

# Sub-linear Time Support Recovery for Compressed Sensing using Sparse-Graph Codes

Xiao Li, Sameer Pawar and Kannan Ramchandran\*

Department of Electrical Engineering and Computer Science (EECS)

University of California, Berkeley

{xiaoli, spawar, kannanr}@eecs.berkeley.edu

December 25, 2014

## Abstract

We address the problem of robustly recovering the support of high-dimensional sparse signals<sup>1</sup> from linear measurements in a low-dimensional subspace. We introduce a new family of sparse measurement matrices associated with low-complexity recovery algorithms. Our measurement system is designed to capture observations of the signal through *sparse-graph codes*, and to recover the signal by using a simple peeling decoder. As a result, we can simultaneously reduce both the measurement cost and the computational complexity. In this paper, we formally connect general sparse recovery problems in compressed sensing with sparse-graph decoding in packet-communication systems, and analyze our design in terms of the measurement cost, computational complexity and recovery performance.

Specifically, in the noiseless setting, our scheme requires  $2K$  measurements asymptotically to recover the sparse support of any  $K$ -sparse signal with  $O(K)$  arithmetic operations. In the presence of noise, both measurement and computational costs are  $O(K \log^{1.3} N)$  for recovering any  $K$ -sparse signal of dimension  $N$ . When the signal sparsity  $K$  is sub-linear in the signal dimension  $N$ , our design achieves *sub-linear time support recovery*. Further, the measurement cost for noisy recovery can also be reduced to  $O(K \log N)$  by increasing the computational complexity to near-linear time  $O(N \log N)$ . In terms of recovery performance, we show that the support of any  $K$ -sparse signal can be stably recovered under finite signal-to-noise ratios with probability one asymptotically.

---

\*This work was supported by grants NSF CCF EAGER 1439725, and NSF CCF 1116404 and MURI CHASE Grant No. 556016.

<sup>1</sup>In this paper, the signal of interest can be sparse with respect to any known basis.

# 1 Introduction

A classic problem of interest in many applications is that of estimating an unknown vector  $\mathbf{x}$  of length  $N$  from noisy observations

$$\mathbf{y} = \mathbf{B}\mathbf{x} + \mathbf{w}, \quad (1)$$

where  $\mathbf{B}$  is an  $M \times N$  known matrix typically referred to as the *measurement matrix* and  $\mathbf{w}$  is an additive noise vector. We refer to  $N$  as the signal dimension. In general, if  $\mathbf{x}$  has no additional structure, it is impossible to recover  $\mathbf{x}$  from fewer measurements than the signal dimension. However, if the signal is known to be sparse in some basis, wherein only  $K$  coefficients are non-zero or significant with  $K \ll N$ , it is possible to recover the signal from significantly fewer measurements. This has been studied extensively in the literature under the name of *compressed sensing* [1]. The compressed sensing problem of reconstructing high-dimensional signals from lower dimensional observations arises in diverse fields, such as medical imaging [2], optical imaging [3], speech and image processing [4], data streaming and sketching [5] etc. By exploiting the inherent signal sparsity, researchers in different fields have developed their own designs to reconstruct sparse signals from low dimensional linear measurements (see Section 2.2 for a brief review of these methods).

Our design takes a “divide-and-conquer” approach by viewing compressed sensing through a new “sparse-graph coding” lens. Our design philosophy is depicted in Fig. 1 as a cartoon illustration, where we use different colors to distinguish different entries in the sparse vector, namely, we choose *red*, *green* and *blue* respectively for the non-zero entries, and *white* for zero entries. A conventional design in compressed sensing is to generate weighted linear measurements of the sparse vector through a carefully designed *measurement matrix* [6]. In this example, all the entries of the measurement matrix are colored in *grey* to indicate some arbitrary design and the corresponding measurements are some generic mixtures of *red*, *green* and *blue*, as shown in Fig. 1-(a).

We design the measurement matrix by sparsifying each row of the measurement matrix with zero patterns guided by sparse-graph codes, indicated by the *white* spots in Fig. 1-(b). This new measurement matrix leads to a different set of measurements, where some contain single colors and some contain their mixtures. Our design philosophy is to *disperse* the signal into multiple single color measurements (e.g., the red color in the first measurement) and *peel* them off from color mixtures (e.g., the red-blue mixture in the second measurement and the blue-green mixture in the third measurement) to decode other unknown colors in the spirit of “divide-and-conquer”. By analogy, the use of sparse-graph codes essentially *divides* the general sparse recovery problem into multiple sub-problems that can be easily *conquered* and synthesized for reconstructions. Furthermore, by viewing our design from a coding-theoretic lens, our design can further leverage the properties of sparse-graph codes in terms of both measurement cost (near-capacity-achieving) and computational complexity (fast peeling-based decoding). This leads to a new family of sparse measurement matrices simultaneously featuring low measurement costs and low computational costs.

## 1.1 Objective

In this paper, we focus on the recovery of the *exact support* of any  $K$ -sparse  $N$ -length signal. This so-called *support recovery* problem arises in an array of applications such as model selection [7], group testing [8], sparse approximation [9] and subset selection in regression problems [10]. Given  $\hat{\mathbf{x}}$  generated by some recovery method, there are various criteria for evaluating the recovery performance. A typical metric for *support recovery* is the probability  $\mathbb{P}_F$  of failing to recover the *exact support* of the signal,

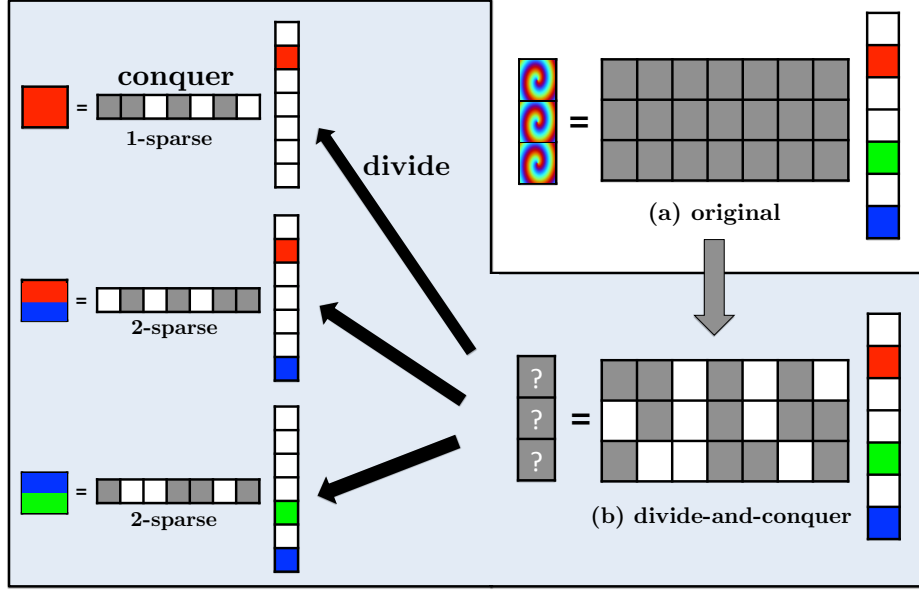


Figure 1: A conceptual cartoon diagram of the “divide-and-conquer” philosophy used in our design. In this diagram, zero entries are colored in *white* and the non-zero entries in the sparse vector are colored in *red*, *green* and *blue* respectively. We have a 3-sparse recovery problem in sub-figure (a), where the measurement matrix is colored in *grey* to indicate an arbitrary design. The resulting measurements are colored as *mixtures* because of the arbitrary mixing of different color components (red, green, blue). In sub-figure (b), we sparsify the measurement matrix by introducing three zeros in each row shown as the *white* spots. The resulting sparsified measurement matrix effectively divides the 3-sparse recovery problem into multiple sub-problems, where one of the sub-problems involves only one color that can be easily identified. In this example, the first measurement contains a single *red* color, whereas the second and third measurements contain a *mixture of red and blue* and a *mixture of blue and green* respectively. If the decoder knows that the first measurement contains a single *red* color, it can peel off its contribution from the *mixture of red and blue* in the second measurement, which forms a new measurement containing a single *blue* color. Similarly, the blue color can be peeled off from the third measurement such that the *green* color is decoded.

defined as

$$\mathbb{P}_F := \Pr (\text{supp} (\hat{\mathbf{x}}) \neq \text{supp} (\mathbf{x})) , \quad (2)$$

where  $\text{supp} (\cdot)$  represents the support of some vector

$$\text{supp} (\mathbf{x}) := \{k : x[k] \neq 0, k \in [N]\} . \quad (3)$$

where  $[N]$  is the set of integers  $\{0, 1, \dots, N - 1\}$ . The probability  $\mathbb{P}_F$  is evaluated with respect to the randomness associated with the noise  $\mathbf{w}$  and the measurement matrix  $\mathbf{B}$ . In other words, for any given  $K$ -sparse signal  $\mathbf{x}$ , our design generates a measurement matrix  $\mathbf{B}$  (from a specific random ensemble<sup>2</sup>) and produces an estimate  $\hat{\mathbf{x}}$  whose support matches *exactly* that of  $\mathbf{x}$  with probability  $1 - \mathbb{P}_F$  approaching one asymptotically in  $K$  and  $N$ .

<sup>2</sup>Note that this is what is known as the “for-each” guarantee [5] in contrast to the “for-all” guarantee in some compressed sensing contributions, where a single measurement matrix is used for all signals once generated.

## 1.2 Our Contributions

Our key contribution is the proposed new compressed sensing design for support recovery featuring *sub-linear time* complexity<sup>3</sup> and near-optimal measurement costs. The sub-linear time feature can potentially enable real-time or near-real-time processing for massive datasets featuring sparsity, which are relevant to a multitude of practical applications. Here we briefly summarize our technical results in terms of measurement and computational costs. As stated in Section 2, in the noiseless setting, our design uses  $2K$  measurements asymptotically with  $O(K)$  arithmetic operations<sup>4</sup>. In the presence of noise, when the sparsity  $K$  is sub-linear in the signal dimension  $N$ , our scheme achieves sub-linear costs on both measurements  $O(K \log^{1.3} N)$  and computations  $O(K \log^{1.3} N)$ . Our proposed framework also admits the option of using  $O(K \log N)$  measurements by increasing the computational cost to near-linear time  $O(N \log N)$ .

We now provide some intuition about our sub-linear time results. Recall that the idea is to use sparse-graph codes to structure the measurement matrix in order to generate subsets of measurements containing isolated 1-sparse coefficients, as well as their mixtures. From Fig. 1, these 1-sparse coefficients (e.g., the red color in the first measurement) can be peeled off from their mixtures (e.g., the red and blue mixture in the second measurement), which forms new 1-sparse coefficients for further peeling. This divide-and-conquer approach allows us to tackle a  $K$ -sparse recovery problem by solving a series of 1-sparse problems instead. Clearly, the challenge is to keep this peeling process going until all 1-sparse components have been recovered. Therefore, we invoke sparse-graph codes principles to study this “turbo” peeling process theoretically to guarantee the success of decoding. As a result, we can focus on solving each 1-sparse problem. Clearly, depending on the specific measurement matrix used, there are many ways to solve these 1-sparse recovery problems. In this paper, our sub-linear results are based on exploiting the Discrete Fourier Transform (DFT) matrix and leveraging spectral estimation techniques [11]. Suppose that we choose the first two rows of the DFT matrix as the measurement matrix before being sparsified by sparse-graph codes. We have two measurements to estimate the unknown index and the unknown value of the 1-sparse coefficient, which is equivalent to estimating the frequency and amplitude of a complex discrete sinusoid from the DFT matrix. Therefore, in the noiseless setting, this can be done by simply examining the relative phase between the two measurements, which only requires  $O(1)$  measurements and computations. In the noisy setting, we further devise a successive spectral estimation scheme using only  $O(\log^{1.3} N)$  measurements and  $O(\log^{1.3} N)$  operations. Finally, since there are in total  $K$  sparse coefficients to estimate, the overall measurement and computational costs become  $O(K)$  for the noiseless setting and  $O(K \log^{1.3} N)$  for the noisy setting.

## 1.3 Notation and Organization

Throughout this paper, we use  $\mathbb{R}$  and  $\mathbb{C}$  to denote the real and complex fields. Any boldface lowercase letter such as  $\mathbf{x} \in \mathbb{C}^N$  represents a vector containing the complex samples  $\mathbf{x} = [x[0], \dots, x[N-1]]^T$ , and a boldface uppercase letter, such as  $\mathbf{X} \in \mathbb{C}^{M \times N}$ , represents a matrix with elements  $X_{i,j}$  for  $i \in [M]$  and  $j \in [N]$ . The calligraphic uppercase letter (i.e.,  $\mathcal{A}$ ) represents a set with cardinality denoted by  $|\mathcal{A}|$ , and the complement of set  $\mathcal{A}$  is denoted by  $\mathcal{A}^c$ . The inner product between two vectors is defined as  $\langle \mathbf{x}, \mathbf{y} \rangle = \sum_{t \in [N]} x[t](y[t])^*$  with arithmetic over  $\mathbb{C}$ .

<sup>3</sup>The computational costs in this paper refer to the reconstruction time after *linear measurements* are available.

<sup>4</sup>Recall that a single variable function  $f(x)$  is said to be  $O(g(x))$ , if for a sufficiently large  $x$  the function  $|f(x)|$  is bounded above by  $|g(x)|$ , i.e.,  $\lim_{x \rightarrow \infty} |f(x)| < c|g(x)|$  for some constant  $c$ . Similarly,  $f(x) = \Omega(g(x))$  if  $\lim_{x \rightarrow \infty} |f(x)| > c|g(x)|$  and  $f(x) = o(g(x))$  if the growth rate of  $|f(x)|$  as  $x \rightarrow \infty$ , is negligible as compared to that of  $|g(x)|$ , i.e.  $\lim_{x \rightarrow \infty} |f(x)|/|g(x)| = 0$ .

This paper is organized as follows. In Section 2.2, we provide a brief overview of existing sparse recovery methods. Then we briefly summarize our main technical results in Section 2. In Section 3, for illustration purpose we provide a concrete example of our design framework using sparse-graph codes, followed by the analysis of the peeling decoder for sparse support recovery. Based on the example, we propose the principle and mathematical formulation of our measurement design in Section 4. Then, we proceed to discuss specific constructions for our noiseless recovery results in Section 6 and further the noisy recovery results in Section 7. Last but not least, we provide numerical results in Section 8 to corroborate our noiseless and noisy recovery performance.

## 2 Main Results and Related Work

In this paper, we consider the problem of recovering the sparse support of  $\mathbf{x}$  from the measurements<sup>5</sup> obtained in (1). In particular, we are interested in support recovery for the noiseless and noisy settings. Our design is characterized by the triplet  $(M, T, \mathbb{P}_F)$ , where  $M$  is the measurement cost,  $T$  is the computational complexity in terms of complex additions and multiplications, and  $\mathbb{P}_F$  is the failure probability defined in (2).

### 2.1 Main Results

Assuming that the support of the sparse signal is arbitrary and the sparsity  $K = o(N)$  is sub-linear with respect to the signal dimension  $N$ , we propose three designs that achieve different operating points  $(M, T, \mathbb{P}_F)$  as follows

1. Noiseless recovery scheme with sub-linear time<sup>6</sup>;
2. Noisy recovery scheme with near-linear time<sup>7</sup>;
3. Noisy recovery scheme with sub-linear time.

The technical results of these designs are summarized below.

**Theorem 1** (Sub-linear Time Noiseless Recovery). *In the absence of noise  $\mathbf{w} = \mathbf{0}$ , given any  $K$ -sparse signal  $\mathbf{x}$  with  $x[k] \in \mathbb{C}$  for  $k \in \text{supp}(\mathbf{x})$ , our noiseless recovery scheme achieves*

1. a measurement cost  $M = 2K(1 + \epsilon)$  for any  $\epsilon > 0$ ;
2. a computational cost  $T = O(K)$ ;
3. a vanishing failure probability  $\mathbb{P}_F \rightarrow 0$  asymptotically in  $K$  and  $N$ .

*Proof.* The design for noiseless recovery is given in details in Section 6. □

For the noisy settings, we further assume that all the non-zero coefficients belong to a set  $\mathcal{X} = \{Ae^{i\theta} : A \in \mathcal{A}, \theta \in \Theta\}$  where  $\mathcal{A} := \{A_{\min} + \ell\rho\}_{\ell=0}^{L_1-1}$  for some arbitrarily large but finite  $L_1 > 1$ ,  $\rho > 0$  and  $A_{\min} > 0$  while  $\Theta := \{2\pi\ell/L_2\}_{\ell=0}^{L_2-1}$  for some arbitrarily large but finite  $L_2 > 0$ .

<sup>5</sup>More generally, we also allow the signal to be sparse in any linear transform domain. If the signal is sparse in the transform domain, one can pre-multiply the matrix  $\mathbf{B}$  on right by the appropriate inverse transform.

<sup>6</sup>By sub-linear time we mean that the overall computational complexity in terms of arithmetic operations grows at a slower rate than the problem dimension  $N$ , i.e.  $T = o(N)$ .

<sup>7</sup>By near-linear time we mean that the overall computational complexity in terms of arithmetic operations grows at a faster rate than  $N$ , i.e.  $T = \Omega(N)$  but slower rate than  $N^{1+\delta}$  for any  $\delta > 0$ , i.e.  $T = o(N^{1+\delta})$ .

**Theorem 2** (Near-linear Time Noisy Recovery). *In the presence of circularly complex Gaussian noise  $\mathbf{w} \sim \mathcal{CN}(\mathbf{0}, \sigma^2 \mathbf{I})$ , given any  $K$ -sparse signal  $\mathbf{x}$  with  $x[k] \in \mathcal{X}$  for  $k \in \text{supp}(\mathbf{x})$ , our noisy recovery scheme with near-linear time achieves*

1. a measurement cost of  $M = O(K \log N)$ ;
2. a computational cost of  $T = O(N \log N)$ ;
3. a vanishing failure probability  $\mathbb{P}_F \rightarrow 0$  asymptotically in  $K$  and  $N$ .

*Proof.* The design for our near-linear time noisy recovery scheme is given in details in Section 7.  $\square$

**Theorem 3** (Sub-linear Time Noisy Recovery). *In the presence of circularly complex Gaussian noise  $\mathbf{w} \sim \mathcal{CN}(\mathbf{0}, \sigma^2 \mathbf{I})$ , given any  $K$ -sparse signal  $\mathbf{x}$  with  $x[k] \in \mathcal{X}$  for  $k \in \text{supp}(\mathbf{x})$ , our noisy recovery scheme with sub-linear time achieves*

1. a measurement cost of  $M = O(K \log^{1.3} N)$ ;
2. a computational cost of  $T = O(K \log^{1.3} N)$ ;
3. a vanishing failure probability  $\mathbb{P}_F \rightarrow 0$  asymptotically in  $K$  and  $N$ .

*Proof.* The design for our sub-linear time noisy recovery scheme is given in details in Section 7.  $\square$

## 2.2 Related Work

In this section, we provide a summary of sparse recovery methods and point out a few differences that distinguish our design from existing results that can be roughly categorized into four classes. In the following we provide a brief overview of each class and place our design in context.

*Convex Relaxation Approach:* The classic formulation for sparse recovery from linear measurements is through an  $\ell_0$ -norm minimization, which is a non-convex optimization problem. This problem has been known to be notoriously hard to solve. Convex optimization techniques relax the original combinatorial problem to a convex  $\ell_1$ -norm minimization problem, where computationally efficient algorithms are designed to solve this relaxed problem. It has been shown that as long as the measurement matrices satisfy the Restricted Isometry Property (RIP) or mutual coherence (MC) conditions, the  $\ell_1$ -relaxation of the original problem has exactly the same sparse solution as the original combinatorial problem. This class of methods is known to provide a high level of robustness against the measurement noise, and furthermore, do not depend on the structure of measurement matrices. Popular algorithms in this class include LASSO [12], Iterative Hard Thresholding (IHT) [13], fast iterative shrinkage-thresholding algorithm (FISTA) [14], message passing [15], Dantzig selector [16] and so on. Most of the existing results along this line measurement matrices that are characterized by a measurement cost of  $O(K \log(N/K))$  and a computational complexity  $O(\text{poly}(N))$ .

*Greedy Methods:* Another class of methods, referred to as greedy iterative algorithms, attempt to solve the original  $\ell_0$ -minimization problem directly using successive approximations of the sparse signal through various heuristics. Examples include Orthogonal Matching Pursuit (OMP) [17], CoSaMP [18], Regularized OMP (ROMP) [19], Stagewise OMP (StOMP) [20] and so on. Similar to convex relaxation approaches, this class also does not depend on the structure of the measurement. Although greedy algorithms are generally faster in practical implementations than the techniques based on convex relaxations, the common computational cost still scales as  $O(\text{poly}(N))$  for both noiseless and noisy settings, with a

few exceptions that incur near-linear run-time  $O(N \log N)$  (e.g., StOMP algorithm [20]). Besides, the measurement matrix is typically stated in terms of MC conditions which require<sup>8</sup>  $O(K^2)$  measurements.

*Coding-theoretic Approach:* This class of methods borrows insights from coding theory to facilitate measurement designs and sparse recovery. For instance, [23, 24] use algebraic codes and [25] uses a generalization of Reed-Solomon codes for measurement designs. There are also recovery algorithms based on list decoding [26, 27], efficient erasure-correcting codes [28–30] and approximate message passing using spatially-coupled LDPC codes [31]. Meanwhile, recovery algorithms based on *expander graphs* [32, 33] achieve near-linear time<sup>9</sup> recovery  $O(N \log(N/K))$  using  $O(K \log(N/K))$  measurements in the noiseless setting. Motivated by expander-based designs, researchers have proposed greedy approximation schemes that achieve similar costs, such as Expander Matching Pursuit (EMP) [35] and Sparse Matching Pursuit (SMP) [36]. Our design shares some similar elements with this class of methods in terms of the code properties being used. However, our approach differs significantly in terms of the measurement design and decoding algorithm analysis, which calls for tools from modern coding theory (e.g., density evolution).

*Group Testing and Sketching:* This class of methods exploit linear “sketches” of data for sparsity pattern recovery in *group testing* [8] and *data stream computing* [37], which pre-dates the field of compressed sensing. In group testing, the common scenario is that we need to devise a collection of tests to find  $K$  anomalous items from  $N$  total items, where the typical goal is to recover the *support* of the underlying sparse vector and minimize the number of tests performed (measurements taken) [38]. On the other hand, the goal of data stream computing is to maintain a short linear sketch of the network flows for approximating the sparse vector with some distortion measure. Examples include the count-min/count-sketch methods [39, 40] and so on. Typical results in this bulk of literature require  $O(K \log(N/K))$  measurements and near-linear time  $O(N \log N)$  (see [5]). While there is a subset of sketching algorithms that achieve sub-linear time with  $O(K \log^{O(1)} N)$  operations [41–43], these results typically tackle noiseless<sup>10</sup> measurements.

Further, with a few exceptions, most of these sparse recovery results have been predominantly developed under the  $\ell_2/\ell_1$ -norm or  $\ell_1/\ell_1$ -norm approximation error metrics<sup>11</sup>, with a relatively much lower coverage of support recovery [5, 16, 22, 44, 45]. Necessary and sufficient conditions for support recovery have been studied in different regimes under various distortion measures using optimal decoders [46–50],  $\ell_1$ -minimization methods [7, 51] and greedy methods [52]. For example, it is shown in [48] that  $O(K \log(N/K))$  measurements are sufficient and necessary for support recovery when the measurement matrix consists of independent identically distributed (i.i.d.) Gaussian entries under Gaussian noise. Similar conditions under other signal and measurement models are also reported in [53–55]. Nonetheless, constructive recovery schemes that specifically target *support recovery* are relatively scarce [40, 54, 56]. In the following, we present the main idea of how to achieve our sub-linear time support recovery with near-optimal measurements in details.

---

<sup>8</sup>Achieving a scaling of  $O(K \log(N/K))$  for greedy pursuit methods requires analyses based on RIP [21]. While there are some results on this topic, it is still ongoing work (see [22]).

<sup>9</sup>Using the same measurement design based on expanders,  $\ell_1$ -minimization can also be shown to achieve similar performance in polynomial time [34].

<sup>10</sup>Although sketching algorithms are not derived specifically to address noisy measurements, they could potentially be quite robust to various forms of noise.

<sup>11</sup> $\ell_p/\ell_q$ -norm guarantees refer to the error metrics measured with respect to the best  $K$ -term approximation error  $\|\mathbf{x}_K - \mathbf{x}\|$  (i.e., the vector  $\mathbf{x}_K$  is the best  $K$ -term approximation containing the  $K$  most significant entries in the sparse vector  $\mathbf{x}$ ), where the recovered sparse signal  $\hat{\mathbf{x}}$  satisfies  $\|\hat{\mathbf{x}} - \mathbf{x}\|_p \leq \kappa \|\mathbf{x}_K - \mathbf{x}\|_q$  for some absolute constant  $\kappa > 0$ .

### 3 Main Idea of Compressed Sensing using Sparse-Graph Codes

In this section, we present our design philosophy depicted in Fig. 1 with more details, and describe the main idea of our measurement design and recovery algorithm through a simple example.

#### 3.1 Design Philosophy

As stated in the introduction, our design philosophy is depicted in Fig. 1 as a cartoon illustration, where we use different colors to distinguish different entries in the sparse vector, namely, we choose *red*, *green* and *blue* respectively for the non-zero entries, and *white* for zero entries. A conventional design in compressed sensing is to generate weighted linear measurements of the sparse vector through a carefully designed *measurement matrix* [6], e.g. popularly based on random independent identically distributed (i.i.d.) sub-Gaussian entries. In this example, all the entries of the measurement matrix are colored in *grey* to indicate some arbitrary design and the corresponding measurements are some generic mixtures of *red*, *green* and *blue*, as shown in Fig. 1-(a).

We design the measurement matrix by sparsifying each row of the measurement matrix with zero patterns guided by sparse-graph codes, indicated by the *white* spots in Fig. 1-(b). This new measurement matrix leads to a different set of measurements, where some contain single colors and some contain their mixtures. In this example, the measurements become separately colored with *red*, a *mixture of red and blue* and a *mixture of blue and green* as shown on the left of Fig. 1-(b). If the decoder knows that the first measurement contains a single *red* color, it can peel off its contribution from the *mixture of red and blue* in the second measurement, which forms a new measurement containing a single *blue* color. Similarly, the blue color can be peeled off from the third measurement such that the *green* color is decoded. In other words, our design can efficiently isolate the measurements that contain a single color, iteratively peel them off from their mixtures in other measurements, and continue this process until all the colors in the sparse vector are recovered.

Next, we illustrate the principle of our recovery algorithm by connecting support recovery with sparse-graph decoding using an “oracle” (described below). Then, using the insights gathered from the oracle-based decoding algorithm, we explain how we can get rid of the “oracle” using the same example.

#### 3.2 Oracle-based Sparse-Graph Decoding

Consider a simple illustration consisting of a sparse signal  $\mathbf{x}$  of length  $N = 20$  with  $K = 5$  non-zero coefficients  $x[1] = 1$ ,  $x[3] = 4$ ,  $x[5] = 2$ ,  $x[10] = 3$  and  $x[13] = 7$ . To illustrate the principle of our recovery algorithm, we construct a bipartite graph with 20 left nodes and 9 right nodes. The left and right nodes are referred to as the *variable nodes* and *check nodes* respectively in the language of sparse-graph codes. The graph has the following properties:

- Each *variable node* (left) labeled with  $k$  is assigned a value  $x[k]$  for  $k \in [N]$ ;
- Each *variable node* (right) is connected to the *check nodes* according to the *sparse* bipartite graph<sup>12</sup> in Fig. 2;
- Each *check node* (right) labeled with  $r$  is assigned a value  $y_r$  equal to the complex sum of its left neighbors, similar to the parity-check constraints of the LDPC codes.

<sup>12</sup>Since the values of the check nodes are not affected by the variable nodes carrying zero coefficients, we show only the edges from the variable nodes with non-zero values  $x[k] \neq 0$ .



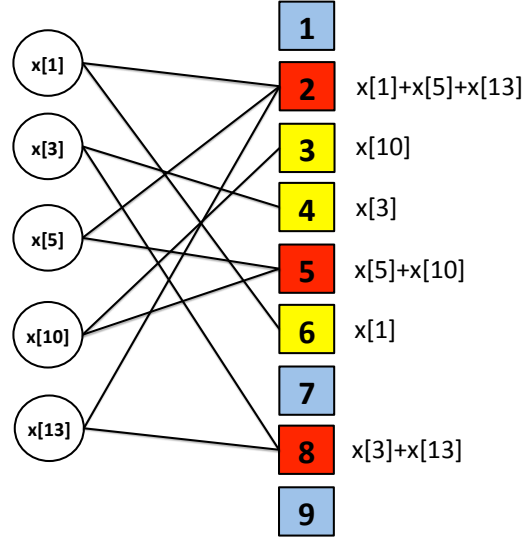


Figure 2: Example of a sparse bipartite graph consisting of 5 (non-zero) left nodes (variable nodes) with 2 edges randomly connected to the right nodes (check nodes). Blue color represents “zero-ton”, yellow color represents “single-ton” and red color represents “multi-ton”.

Now we briefly introduce how this bipartite graph helps us recover the 20-length sparse signal  $\mathbf{x}$  on the variable nodes from the 9 measurements associated with the check nodes:

$$\begin{aligned}
 y_1 &= y_7 = y_9 = 0, \\
 y_2 &= x[1] + x[5] + x[13], \\
 y_3 &= x[10], \\
 y_4 &= x[3], \\
 y_5 &= x[5] + x[10], \\
 y_6 &= x[1], \\
 y_8 &= x[3] + x[13].
 \end{aligned}$$

Depending on the connectivity of the sparse bipartite graph, we categorize the measurements associated with the check nodes into the following types:

1. **Zero-ton:** a check node is a zero-ton if it does not involve any non-zero coefficient (e.g., the color *blue* in Fig. 2).
2. **Single-ton:** a check node is a single-ton if it involves only one non-zero coefficient (e.g., the color in *yellow* in Fig. 2). More specifically, we refer to the index  $k$  of the non-zero coefficient  $x[k]$  and its associated value  $x[k]$  as the **index-value pair**  $(k, x[k])$  for that single-ton.
3. **Multi-ton:** a check node is a multi-ton if its value is the sum of more than one non-zero coefficient (e.g., the color *red* in Fig. 2).

To help illustrate our decoding algorithm, we assume that there exists an “oracle” that informs the decoder exactly which check nodes are *single-tons*. More importantly, the oracle further provides the index-value pair for that single-ton. In this example, the oracle informs the decoder that check nodes labeled 3, 4 and 6 are single-tons with index-value pairs  $(10, x[10])$ ,  $(3, x[3])$  and  $(1, x[1])$  respectively. Then the decoder can subtract their contributions from other check nodes, forming new single-tons.

Therefore generally speaking, with the oracle information, the peeling decoder repeats the following steps similar to [57, 58]:

**Step (1)** select all the edges in the bipartite graph with right degree 1 (identify single-ton bins);

**Step (2)** remove (peel off) these edges as well as the corresponding pair of variable and check nodes connected to these edges.

**Step (3)** remove (peel off) all other edges connected to the variable nodes that have been removed in **Step (2)**.

**Step (4)** subtract the contributions of the variable nodes from check nodes whose edges have been removed in **Step (3)**.

Finally, decoding is successful if all the edges are removed from the graph.

### 3.3 Getting Rid of the Oracle

Since the oracle information is critical in the peeling process, we proceed with our example and explain briefly how to obtain such information without an oracle. Now, as an example, instead of simply assigning the *complex sum* to each check node, we assign a *vector-weighted complex sum* to the check nodes, where each variable node (say  $k$ ) is weighted by the  $k$ -th column of the following matrix

$$\mathbf{S} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & \cdots & 1 \\ 1 & W & W^2 & W^3 & W^4 & \cdots & W^{19} \end{bmatrix},$$

where  $W = e^{i\frac{2\pi}{N}}$  is the  $N$ -th root of unit with  $N = 20$ . Note that this is simply the first two rows of the  $20 \times 20$  DFT matrix. In this way, each check node (say  $m$ ) is assigned a 2-dimensional vector  $\mathbf{y}_r = [y_r[0], y_r[1]]^T$  and we call each vector a **measurement bin**. For example, the measurements associated with check node 1, 2 and 3 become

$$\begin{aligned} \mathbf{y}_1 &= \mathbf{0}, \\ \mathbf{y}_2 &= x[1] \times \begin{bmatrix} 1 \\ W \end{bmatrix} + x[5] \times \begin{bmatrix} 1 \\ W^5 \end{bmatrix} + x[13] \times \begin{bmatrix} 1 \\ W^{13} \end{bmatrix}, \\ \mathbf{y}_3 &= x[10] \times \begin{bmatrix} 1 \\ W^{10} \end{bmatrix}. \end{aligned}$$

Now with these bin measurements, one can effectively determine if a check node is a zero-ton, a single-ton or a multi-ton. Although this procedure is formally stated in Section 6 in our noiseless recovery results, here as an illustration, we go through the procedures for check nodes 1, 2 and 3:

- **zero-ton bin:** consider the zero-ton check node 1. A zero-ton check node can be identified easily since the measurements are all zero

$$\mathbf{y}_1 = \mathbf{0}. \tag{4}$$

- **single-ton bin:** consider the single-ton check node 3. A single-ton can be verified by performing a simple “ratio test” of the two dimensional vector:

$$\begin{aligned} \hat{k} &= \frac{\angle y_3[1]/y_3[0]}{2\pi/20} = 10, \\ \hat{x}[\hat{k}] &= y_3[0] = 3. \end{aligned}$$

Another unique feature is that the measurements would have identical magnitudes  $|y_3[0]| = |y_3[1]|$ . Both the ratio test and the magnitude constraints are easy to verify for all check nodes such that the index-value pair is obtained for peeling.

- **multi-ton bin**: consider the multi-ton check node 2. A multi-ton can be easily identified by the same ratio test

$$\hat{k} = \frac{\angle y_2[1]/y_2[0]}{2\pi/20} = 12.59.$$

Furthermore, the magnitudes are not identical  $|y_2[0]| \neq |y_2[1]|$ . Therefore, if the ratio test does not produce a non-zero integer and the magnitudes are not identical, we can conclude that this check node is a multi-ton.

This simple example shows how the problem of recovering the  $K$ -sparse signal  $\mathbf{x}$  can be cast as an instance of sparse-graph decoding. It also suggests that it is possible to obtain the index-value pair of any single-ton without the help of an “oracle”. Note that the sparse bipartite graph in this example only shows the idea of peeling decoding, but does not guarantee successful recovery for an arbitrary signal. We will later address in Section 5 how to construct sparse bipartite graphs to guarantee successful decoding, but in the following, we first present our general measurement design in Section 4.

## 4 Measurement Design

Before delving into the specifics, we define an operator that facilitates the explanation of our measurement design. Given a matrix  $\mathbf{S} \in \mathbb{C}^{M_2 \times N}$  and a matrix  $\mathbf{H} \in \mathbb{C}^{M_1 \times N}$  with  $M$  rows, each row denoted by  $\mathbf{h}_j^T$  for  $j \in [M_1]$ , the *row-tensor product* is defined as

$$\mathbf{H} \odot \mathbf{S} := \begin{bmatrix} \mathbf{h}_1^T \otimes \mathbf{S} \\ \vdots \\ \mathbf{h}_{M_1}^T \otimes \mathbf{S} \end{bmatrix},$$

where  $\otimes$  is the standard Kronecker product. For example, let  $\mathbf{H}$  be a sparse matrix with random coding patterns of  $\{0, 1\}$  and  $\mathbf{S}$  be chosen as the first two rows of a DFT matrix as in the simple example

$$\mathbf{H} = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}, \quad \mathbf{S} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & W & W^2 & W^3 & W^4 & W^5 & W^6 \end{bmatrix} \quad (5)$$

with  $W = e^{i\frac{2\pi}{7}}$ . Then the row-tensor product is given by

$$\mathbf{H} \odot \mathbf{S} = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & W & 0 & W^3 & 0 & W^5 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & W & 0 & W^3 & 0 & 0 & W^6 \\ 1 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & W^3 & W^4 & W^5 & W^6 \end{bmatrix}. \quad (6)$$

Since  $\mathbf{H}$  has three rows of coding patterns, the product  $\mathbf{H} \odot \mathbf{S}$  contains three blocks of matrices, where each block is the corresponding sparsified version of  $\mathbf{S}$  by the coding pattern in each row of  $\mathbf{H}$ .

**Definition 1 (Measurement Matrix).** Let  $M = RP$  for some positive integers  $R$  and  $P$ . Given a  $R \times N$  coding matrix  $\mathbf{H}$  and a  $P \times N$  bin detection matrix  $\mathbf{S}$ , the  $M \times N$  measurement matrix  $\mathbf{B}$  is given by

$$\mathbf{B} = \mathbf{H} \odot \mathbf{S}, \quad (7)$$

where  $\odot$  is the row-tensor product, and the coding matrix and bin detection matrixs are specified below.

- Given a bipartite graph with
  - $N$  left nodes and  $R$  right nodes,
  - an edge set  $\mathcal{E}$  between the left and right nodes,

the coding matrix  $\mathbf{H} = [H_{r,n}]$  is chosen as the  $R \times N$  bi-adjacency matrix of the bipartite graph

$$H_{r,n} = \begin{cases} 1, & \text{if } \{r, n\} \in \mathcal{E} \\ 0, & \text{if } \{r, n\} \notin \mathcal{E} \end{cases}$$

- $\mathbf{S} := [\mathbf{s}_1, \dots, \mathbf{s}_N]$  is a  $P \times N$  bin detection matrix whose constructions are given in Section 6 on noiseless recovery and Section 7 on noisy recovery.

**Corollary 1.** The measurement  $\mathbf{y}$  is divided into  $R$  measurement bins as  $\mathbf{y} = [\mathbf{y}_1^T, \dots, \mathbf{y}_R^T]^T$  with

$$\mathbf{y}_r = \mathbf{S}\mathbf{z}_r + \mathbf{w}_r, \quad r = 1, \dots, R, \quad (8)$$

where  $\mathbf{w}_r$  is the noise in the  $r$ -th measurement bin,  $\mathbf{z}_r := \text{diag}[\mathbf{h}_r] \mathbf{x}$  and  $\mathbf{h}_r^T$  is the  $r$ -th row<sup>13</sup> of the coding matrix  $\mathbf{H}$ . In the presence of noise  $\mathbf{w}_r$ , the type of each bin is cast as a separate hypothesis:

1.  $\mathbf{y}_r$  is a **zero-ton** bin if  $\text{supp}(\mathbf{z}_r) = \emptyset$ , denoted by  $\mathbf{y}_r \sim \mathcal{H}_Z$ ;
2.  $\mathbf{y}_r$  is a **single-ton** bin with the index-value pair  $(k, x[k])$  if  $\text{supp}(\mathbf{z}_r) = \{k\}$  for some  $k \in [N]$  and  $z[k] = x[k]$ , denoted by  $\mathbf{y}_r \sim \mathcal{H}_S(k, x[k])$ ;
3.  $\mathbf{y}_r$  is a **multi-ton** bin if  $|\text{supp}(\mathbf{z}_r)| \geq 2$ , denoted by  $\mathbf{y}_r \sim \mathcal{H}_M$ .

*Proof.* The proof is straightforward and hence omitted. □

Given the above general measurement design, the following questions are of particular interests:

1. Given  $N$  left nodes and  $R$  right nodes, how to construct a bipartite graph that guarantees a “friendly” distribution of single-tons, zero-tons and multi-tons for successful peeling?
2. Given the sparsity  $K$  such that only  $K$  left nodes remain in the sparse bipartite graph, what is the minimum number of right nodes  $R$  to guarantee successful peeling?
3. How to choose the *bin detection matrix*  $\mathbf{S}$  in general for providing the oracle information? Especially when the measurements are noisy?

In the following, we answer these questions in details and discuss the specific constructions for  $\mathbf{H}$  and  $\mathbf{S}$ . In Section 5, we first present the peeling decoder analysis that guides the design of the bipartite graphs and the associated coding matrix  $\mathbf{H}$ , and then discuss the constructions of the bin detection matrix  $\mathbf{S}$  for both noiseless and noisy scenarios in Section 6 and 7 respectively.

<sup>13</sup>For notational convenience, we used a transpose operator to maintain  $\mathbf{h}_r$  as a column vector.

## 5 Sparse Graph Design and Peeling Decoder

### 5.1 Sparse Graph Design for Compressed Sensing

The design of sparse bipartite graphs for peeling decoders has been studied extensively in the context of erasure-correcting sparse-graph codes [57, 59]. In this section, for simplicity we consider *the ensemble of left  $d$ -regular bipartite graphs*  $\mathcal{G}_{\text{reg}}^N(R, d)$  consisting of  $N$  left nodes and  $R$  right nodes, where each left node  $n \in [N]$  is connected to  $d$  right nodes  $r = 1, \dots, R$  uniformly at random and the number of right nodes is linear in the sparsity  $R = \eta K$ . We call  $\eta$  the *redundancy parameter*.

The coding matrix  $\mathbf{H}$  constructed from the regular graph ensemble conforms with a random “balls-and-bins” model, where each row of  $\mathbf{H}$  corresponds to a “bin” (i.e., right node) and each column of  $\mathbf{H}$  corresponds to a “ball” (i.e., left node). If the  $(r, n)$ -th entry  $H_{r,n} = 1$ , then we say that the  $n$ -th ball is thrown into the  $r$ -th bin. In the “balls-and-bins” model associated with the regular ensemble  $\mathcal{G}_{\text{reg}}^N(R, d)$ , each ball  $n \in [N]$  is thrown uniformly at random to  $d$  bins. In the context of LDPC codes, the  $n$ -th coefficient  $x[n]$  (variable node) appears in the parity check constraints in  $d$  right nodes (check nodes) chosen uniformly at random. For example, consider a smaller example with  $N = 8$  left nodes and  $R = 5$  nodes, where  $\mathbf{x} = [x[0], \dots, x[7]]^T$  is some generic signal vector. Then, an instance from the 2-regular ensemble  $\mathcal{G}_{\text{reg}}^8(5, 2)$  and the associated coding matrix  $\mathbf{H}$  are shown in Fig. 3.

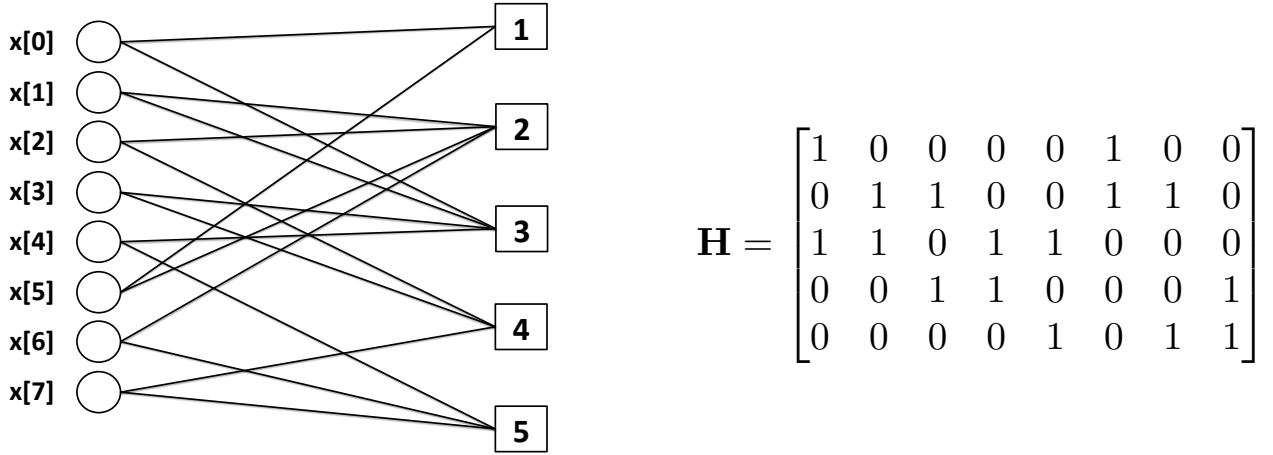


Figure 3: An example of the bipartite graph from the regular graph ensemble with  $d = 2$  left degrees, consisting of  $N = 8$  left nodes and  $R = 5$  nodes, where the left nodes are labeled by the signal  $\mathbf{x} = [x[0], \dots, x[7]]^T$ .

In our compressed sensing design, the sparse bipartite graph for peeling is the “pruned” graph after removing the left nodes with zero values. For example, if the signal is 4-sparse with non-zero coefficients  $x[1]$ ,  $x[4]$ ,  $x[5]$  and  $x[6]$ , then the “pruned” graph is reduced to that in Fig. 4 on the right from the *full graph* on the left. Another example of a “pruned” graph has been shown in Fig. 2, which is associated with a 5-sparse signal and a left 2-regular graph with  $N = 20$  left nodes and  $R = 9$  right nodes.

Clearly, for compressed sensing of an arbitrary  $K$ -sparse signal  $\mathbf{x}$ , the *pruned* graph in Fig. 4, instead of the *full graph* in Fig. 3, determines the peeling decoder performance. However, the pruned graph depicted in Fig. 4 does not lead to successful decoding since the peeling is stuck with all multi-tons after removing the single-ton from right node #1. The intuition is that there are 4 nodes on the left with degree 2 but only 5 nodes on the right, therefore there is a high probability for each right node to connect to more than one left node (i.e., in this case only one right node has degree 1). Therefore, in general, given the left degree  $d$  of the ensemble and the sparsity  $K$ , the graph needs to contain a sufficient number of right nodes to guarantee the success of the peeling decoder by choosing the redundancy parameter  $\eta$

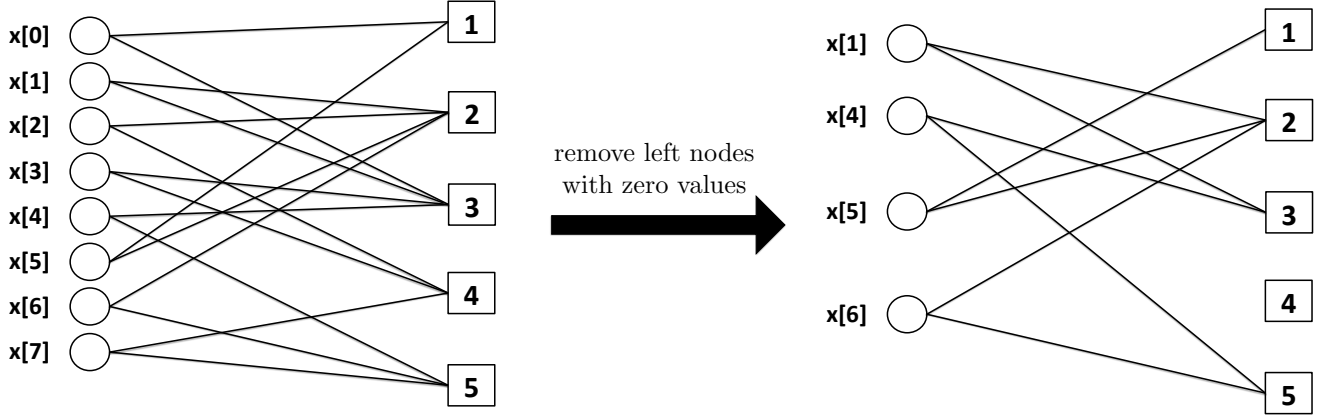


Figure 4: The “pruned” bipartite graph when the signal  $\mathbf{x} = [x[0], \dots, x[7]]^T$  is 4-sparse with non-zero coefficients  $x[1]$ ,  $x[4]$ ,  $x[5]$  and  $x[6]$ .

properly. In the following, we study the peeling decoder performance over the pruned graphs from the regular ensemble  $\mathcal{G}_{\text{reg}}^N(R, d)$  and shed light on how to specify the parameter  $\eta$  appropriately.

## 5.2 Oracle-based Peeling Decoder Analysis using the Regular Ensemble $\mathcal{G}_{\text{reg}}^N(R, d)$

In this section, we show that if the redundancy parameter  $\eta = R/K$  and the left regular degree  $d$  are chosen properly for the regular graph ensemble  $\mathcal{G}_{\text{reg}}^N(R, d)$ , then for an arbitrary  $K$ -sparse signal  $\mathbf{x}$ , all the edges of the *pruned graph* can be peeled off in  $O(K)$  peeling iterations *with high probability*. In other words, we show that as long as the *full graph* is chosen properly, the *pruned graph* can lead to successful decoding with high probability for any given sparse signal. Our analysis is similar to the arguments in [57, 59] using the so-called *density evolution* analysis from modern coding theory, which tracks the average density<sup>14</sup> of the remaining edges in the pruned graph at each peeling iteration of the algorithm.

Although the proof techniques are similar to those from [59] and [57], the graph used in our peeling decoder is different from those in [57, 59]. This leads to fairly important differences in the analysis, such as the degree distributions of the graphs (explained later) and the expansion properties of the graphs. Hence, we present an independent analysis here for our peeling decoder. In the following, we provide a brief outline of the proof elements highlighting the main technical components.

- **Density evolution:** We analyze the performance of our peeling decoder over a *typical graph* (i.e., cycle-free) of the ensemble  $\mathcal{G}_{\text{reg}}^N(R, d)$  for a fixed number of peeling iterations  $i$ . We assume that a local neighborhood of every edge in the graph is cycle-free (tree-like) and derive a recursive equation that represents the average density of remaining edges in the pruned graph at iteration  $i$ .
- **Convergence to density evolution:** Using a Doob martingale argument as in [57] and [60], we show that the local neighborhood of most edges of a randomly chosen graph from the ensemble  $\mathcal{G}_{\text{reg}}^N(R, d)$  is cycle-free with high probability. This proves that with high probability, our peeling decoder removes all but an arbitrarily small fraction of the edges in the pruned graph (i.e., the left nodes are removed at the same time after being decoded) in a constant number of iterations  $i$ .

<sup>14</sup>The density here refers to fraction of the remaining edges, or namely, the number of remaining edges divided by the total number of edges in the graph.

- **Graph expansion property** for complete decoding: We show that if the sub-graph consisting of the remaining edges is an “expander” (as will be defined later in this section), and if our peeling decoder successfully removes all but a sufficiently small fraction of the left nodes from the pruned graph, then it removes all the remaining edges of the “pruned” graph successfully. This completes the decoding of all the non-zero coefficients in  $\mathbf{x}$ .

### 5.2.1 Density Evolution

Density evolution, a powerful tool in modern coding theory, tracks the average density of remaining edges that are not decoded after a fixed number of peeling iteration  $i > 0$ . We introduce the concept of *directed neighborhood* of a certain edge in the pruned graph up to depth  $\ell = 2i$ . This concept is important in the density evolution analysis since the peeling of an edge in the  $i$ -th iteration depends solely on the removal of the edges from this neighborhood in the previous  $i - 1$  iterations. The *directed neighborhood*  $\mathcal{N}_e^\ell$  at depth  $\ell$  of a certain edge  $e = (v, c)$  is defined as the induced sub-graph containing all the edges and nodes on paths  $e_1, \dots, e_\ell$  starting at a variable node  $v$  (left node) such that  $e_1 \neq e$ . An example of a directed neighborhood of depth  $\ell = 2$  is given in Fig. 5.

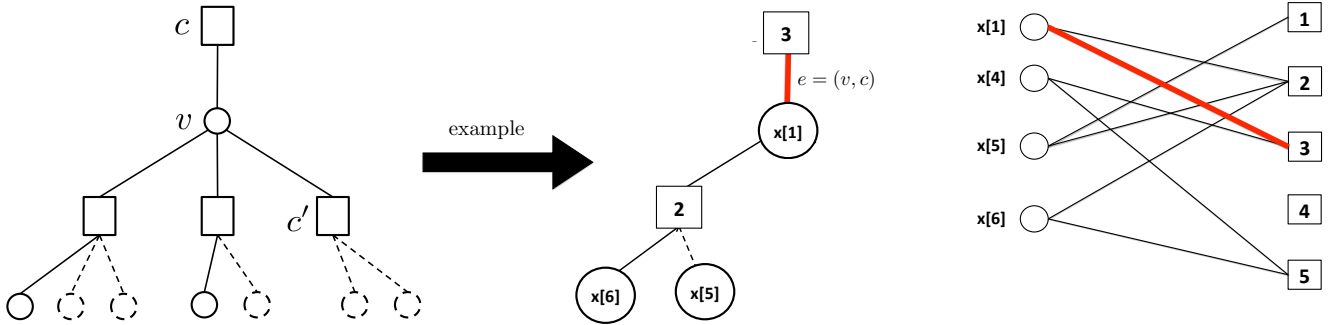


Figure 5: On the left sub-figure, we illustrate the directed neighborhood of depth 2 of an edge  $e = (v, c)$ , namely  $\mathcal{N}_e^2$ , while on the right we show this neighborhood for our example depicted in Fig. 4. The dashed lines on the left correspond to nodes/edges removed at the end of iteration  $i$ . The edge between  $v$  and  $c$  can be potentially removed at iteration  $i + 1$  as one of the check nodes (right nodes)  $c'$  is a single-ton (it has no more variable nodes remaining at the end of iteration  $i$ ). In our example, unlike the check node  $c'$  on the left, the edge  $e = (x[1], 3)$  cannot be removed since the check node is still a multi-ton (i.e.,  $x[6]$  and  $x[1]$  are still attached).

To analyze the performance of the peeling decoder over the pruned graph, we need to understand the edge degree distributions on the left and right for the pruned graph. Let  $\rho_j$  be the fraction of edges in the pruned graph connecting to right nodes with degree  $j$ . Clearly, the total number of edges is  $Kd$  in the pruned graph since there are  $K$  left nodes in the pruned graph and each left node has degree  $d$ . Therefore, since the expected number of right nodes with degree  $j$  can be obtained as  $\Pr(\text{a right node has degree } j) Rj$ , the fraction  $\rho_j$  can be obtained as

$$\rho_j = \frac{\Pr(\text{a right node has degree } j) Rj}{Kd} = \frac{j\eta}{d} \Pr(\text{a right node has degree } j), \quad (9)$$

where we have used  $R = \eta K$  and  $\eta$  is the redundancy parameter. According to the “balls-and-bins” model, the degree of a right node follows the binomial distribution  $B(d/\eta K, K)$  and can be well approximated by a Poisson variable as

$$\Pr(\text{a right node has degree } j) \approx \frac{(d/\eta)^j e^{-d/\eta}}{j!}. \quad (10)$$

As a result, the fraction  $\rho_j$  of edges connected to right nodes having degree  $j$  is

$$\rho_j = \frac{(d/\eta)^{j-1} e^{-d/\eta}}{(j-1)!}. \quad (11)$$

Now let us consider the local neighborhood  $\mathcal{N}_e^{2i}$  of an arbitrary edge  $e = (v, c)$  with a left regular degree  $d$  and right degree distribution given by  $\{\rho_j\}_{j=1}^K$ . If the sub-graph corresponding to the neighborhood  $\mathcal{N}_e^{2i}$  of the edge  $e = (v, c)$  is a *tree* or namely *cycle-free*, then the peeling procedures over different bins in the first  $i$  iterations (see Section 3.2) are independent, which can greatly simplify our analysis. Density evolution analysis is based on the assumption that this neighborhood is cycle-free (tree-like), and we will prove later (in the next subsection) that all graphs in the regular ensemble behave like a tree when  $N$  and  $K$  are large and hence the actual density evolution concentrates well around the density evolution result.

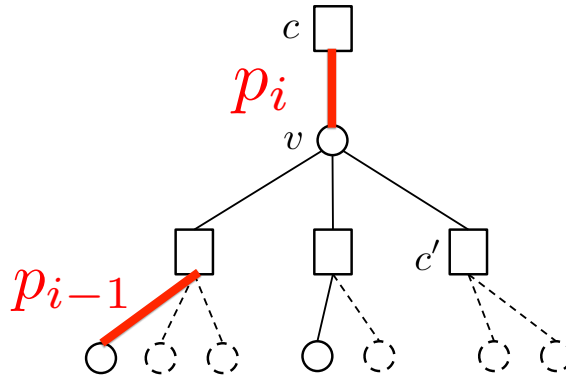


Figure 6: The schematic of density evolution in a local tree-like neighborhood.

Let  $p_i$  be the probability of this edge being present in the pruned graph after  $i > 0$  peeling iterations. If the neighborhood is a tree as in Fig. 6, the probability  $p_i$  can be written with respect to the probability  $p_{i-1}$  at the previous depth in a recursive manner:

$$p_i = \left( 1 - \sum_j \rho_j (1 - p_{i-1})^{j-1} \right)^{d-1}, \quad i = 1, 2, 3, \dots \quad (12)$$

The term  $\sum_j \rho_j (1 - p_{i-1})^{j-1}$  can be approximated using the right degree generating polynomial

$$\rho(x) := \sum_j \rho_j x^{j-1} = e^{-(1-x)\frac{d}{\eta}}, \quad (13)$$

where we have used (11) to derive the second expression. Therefore, the density evolution equation for our peeling decoder is obtained as

$$p_i = f(p_{i-1}) = \left( 1 - e^{-\frac{d}{\eta} p_{i-1}} \right)^{d-1}, \quad i = 1, 2, 3, \dots \quad (14)$$

An example of the density evolution with  $d = 3$  and different values of  $\eta$  is given in Fig. 7. Clearly, the probability  $p_i$  can be made arbitrarily small for a sufficiently large but finite  $i > 0$  as long as  $d$  and  $\eta$  are chosen properly. One can find the minimum value  $\eta$  for a given  $d$  to guarantee  $p_i < p_{i-1}$ , which is shown in Table 1. Due to lack of space we only show up to  $d = 6$ .



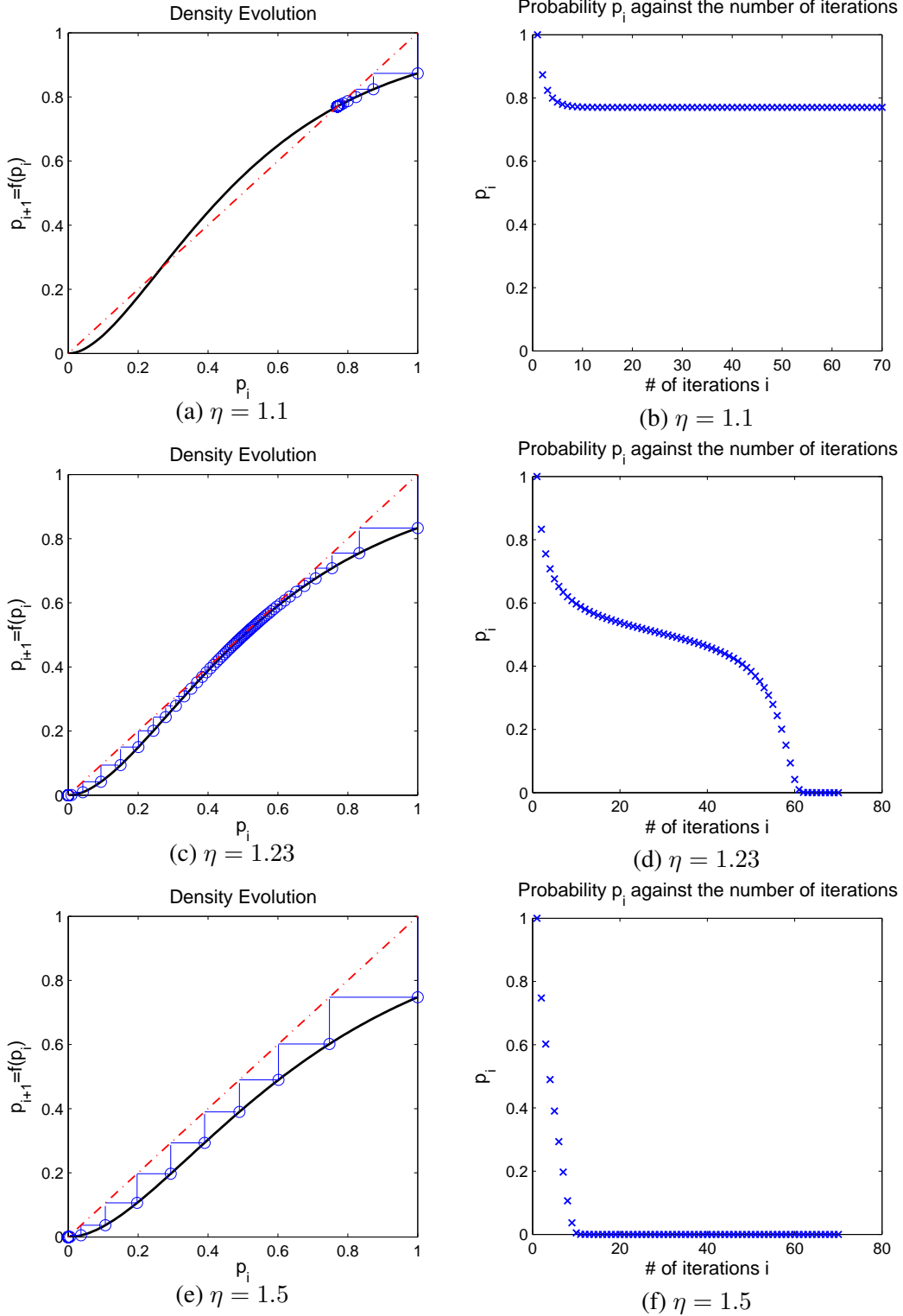


Figure 7: The density evolution  $f(p_i)$  and the probability  $p_i$  at each iteration  $i$ , where we have shown the case with  $d = 3$  and  $\eta = 1.1, \eta = 1.23, \eta = 1.5$ . In the density evolution figures (a)-(c)-(e), the red line is the line  $p_{i+1} = p_i$  while the black line is the actual density evolution recursion  $f(p_i)$  against  $p_i$ . The blue circles that “zig-zag” between the red line and the black line are the specific  $p_i$ ’s that are achieved at each peeling iteration. It can be seen from (a) that when  $\eta$  is small (i.e.  $\eta = 1.1$ ), the density evolution reaches a fixed point at around  $p_i \approx 0.8$ . On the other hand, when  $\eta$  is greater than the threshold 1.23 given by Table 1, the density  $p_i$  reaches 0 very quickly in (a) when  $\eta = 1.5$ . The values of  $p_i$  marked by the blue circles in (a)-(c)-(e) are further plotted against the peeling iterations  $i$  in (b)-(d)-(f), where in the case with  $\eta = 1.5$  the density  $p_i$  approaches 0 after less than 10 iterations.

$d$	2	3	4	5	6
minimum $\eta$	2.0000	1.2219	1.2948	1.4250	1.5696

Table 1: Minimum value for  $\eta$  given the regular degree  $d$  according to density evolution.

**Lemma 1** (Density evolution). *Denote by  $\mathcal{T}_i$  the event where the local  $2i$ -neighborhood  $\mathcal{N}_e^{2i}$  of every edge in the graph is tree-like and let  $Z_i$  be the total number of edges that are not decoded after  $i$  (an arbitrarily large but fixed) peeling iterations. For any  $\varepsilon > 0$ , there exists a finite number of iteration  $i > 0$  such that*

$$\mathbb{E}[Z_i|\mathcal{T}_i] = Kd\varepsilon/4, \quad (15)$$

where the expectation is taken with respect to the random graph ensemble  $\mathcal{G}_{\text{reg}}^N(R, d)$  with the left regular degree  $d$  and the redundancy parameter  $\eta = R/K$  chosen from Table 1.

Based on this lemma, we can see that if the pruned bipartite graph has a local neighborhood that is tree-like up to depth  $2i$  for every edge, the peeling decoder on average peels off all but an arbitrarily small fraction of the edges in the graph. We prove this lemma below.

*Proof.* Let  $Z_i^e \in \{0, 1\}$  be the random variable denoting the presence of edge  $e$  after  $i$  iterations, thus

$$Z_i = \sum_{e=1}^{Kd} Z_i^e. \quad (16)$$

Since each edge is peeled off independently given the event  $\mathcal{T}_i$ , the expected number of remaining edges over cycle-free graphs can be obtained as

$$\mathbb{E}[Z_i|\mathcal{T}_i] = \sum_{e=1}^{Kd} \mathbb{E}[Z_i^e|\mathcal{T}_i] = Kdp_i, \quad (17)$$

where by definition  $p_i = \Pr(Z_i^e = 1|\mathcal{T}_i)$  is the *conditional probability* of an edge in the  $i$ -th peeling iteration conditioned on the event  $\mathcal{T}_i$  studied in the density evolution equation (14). We are interested in the evolution of such probability  $p_i$ . In the following, we prove that for any given  $\varepsilon > 0$ , there exists a finite number of iterations  $i > 0$  such that  $p_i \leq \varepsilon/4$ , which leads to our desired result in (15).  $\square$

### 5.2.2 Convergence to density evolution

Given the mean performance analysis (in terms of the number of undecoded edges) over cycle-free graphs through density evolution, now we provide a *concentration analysis* on the number of the undecoded edges  $Z_i$  for any graph from the regular ensemble at the  $i$ -th iteration, by showing that  $Z_i$  converges to the density evolution result.

**Lemma 2.** *Over the probability space of all graphs from  $\mathcal{G}_{\text{reg}}^N(R, d)$ , let  $p_i$  be as given in the density evolution (14). Given any  $\varepsilon > 0$  and a sufficiently large  $K$ , there exists a constant  $c_4 > 0$  such that*

$$(i) \quad \mathbb{E}[Z_i] < Kd\varepsilon/2 \quad (18)$$

$$(ii) \quad \Pr(|Z_i - \mathbb{E}[Z_i]| > Kd\varepsilon/2) \leq 2 \exp\left(-c_4\varepsilon^2 K^{\frac{1}{4i+1}}\right). \quad (19)$$

*Proof.* We refer the readers to the Appendix in our full report available at [62].  $\square$

### 5.2.3 Graph expansion property for complete decoding

From previous analyses, it has already been established that with high probability, our peeling decoder terminates with an arbitrarily small fraction of edges undecoded

$$Z_i < Kd\varepsilon, \quad \forall \varepsilon > 0, \quad (20)$$

where  $d$  is the left degree. In this section, we show that all the undecoded edges can be completely decoded if the sub-graph consisting of the remaining undecoded edges is a “good-expander”. First, we introduce the concept of graph expanders.

**Definition 2** (Graph Expander). *A bipartite graph with  $K$  left nodes and regular left degree  $d$  is called a  $(\varepsilon, 1/2)$ -expander if for all subsets  $\mathcal{S}$  of left nodes with  $|\mathcal{S}| \leq \varepsilon K$ , there exists a right neighborhood of  $\mathcal{S}$  in the graph, denoted by  $\mathcal{N}(\mathcal{S})$ , that satisfies  $|\mathcal{N}(\mathcal{S})| > d|\mathcal{S}|/2$ .*

**Lemma 3.** *Consider the regular graph ensemble  $\mathcal{G}_{\text{reg}}^N(R, d)$ , then the pruned graph resulting from any given  $K$ -sparse signal  $\mathbf{x}$  is a  $(\varepsilon, 1/2)$ -expander with probability at least  $1 - O(1/K)$  for some sufficiently small but constant  $\varepsilon > 0$ .*

*Proof.* We refer the readers to the Appendix in our full report available at [62].  $\square$

Without loss of generality, let the  $Z_i$  undecoded edges be connected to a set of left nodes  $\mathcal{S}$ . Since each left node has degree  $d$ , it is obvious from (20) that  $|\mathcal{S}| \leq K\varepsilon$  with high probability. Note that our peeling decoder fails to decode the set  $\mathcal{S}$  of left nodes if and only if there are no more single-ton right nodes in the neighborhood of  $\mathcal{S}$ . A sufficient condition for all the right nodes in  $\mathcal{N}(\mathcal{S})$  to have at least one single-ton is that the average degree of the right nodes in the set  $\mathcal{N}(\mathcal{S})$  is at most 2, which implies that  $|\mathcal{S}|d/|\mathcal{N}(\mathcal{S})| \leq 2$  and hence  $|\mathcal{N}(\mathcal{S})| \geq |\mathcal{S}|d/2$ . Since we have shown in Lemma 3 that any pruned graph from the regular ensemble  $\mathcal{G}_{\text{reg}}^N(R, d)$  is a  $(\varepsilon, 1/2)$ -expander with high probability such that  $|\mathcal{N}(\mathcal{S})| \geq d|\mathcal{S}|/2$ , there will be sufficient single-tons to peel off all the remaining edges.

**Theorem 4.** *Consider the ensemble  $\mathcal{G}_{\text{reg}}^N(R, d)$  for our construction, the oracle-based peeling decoder peels off all the edges in the pruned graph in  $O(K)$  iterations with probability at least  $1 - O(1/K)$ .*

*Proof.* The failure probability of the oracle-based peeling decoder is bounded by the probability (see (19)) when (20) is not satisfied, and the probability that the pruned graph is not a  $(\varepsilon, 1/2)$ -expander when (20) is satisfied (given by Lemma 3). Both of these events occur with probability zero asymptotically in  $K$ . Last but not least, since there are a total of  $O(K)$  edges in the pruned graph, and there is at least one edge being peeled off in each iteration with high probability, the result follows.  $\square$

## 6 Noiseless Recovery

### 6.1 Measurement Construction

For the noiseless setting, we choose the *bin detection matrix*  $\mathbf{S}$  as

$$\mathbf{S} := \begin{bmatrix} G_0 & \cdots & G_n & \cdots & G_{N-1} \\ G_0 & \cdots & G_n W^n & \cdots & G_{N-1} W^{N-1} \end{bmatrix} \quad (21)$$

where  $W = e^{i\frac{2\pi}{N}}$  is the  $N$ -th root of unity and  $G_n$  for  $n \in [N]$  is a random variable drawn from some continuous distribution. The bin detection matrix is therefore the first 2 rows of the  $N \times N$  DFT matrix

with each column scaled by a random variable. This is similar to the example we used in Section 3.3, except for the random scaling on each column. The coding matrix  $\mathbf{H}$  requires further discussions. If we use the regular graph ensemble  $\mathcal{G}_{\text{reg}}^N(R, d)$  to construct the coding matrix  $\mathbf{H}$ , the measurement cost becomes  $2\eta K$ , since there are  $R = \eta K$  right nodes and each node has *two* measurements from the bin detection matrix  $\mathbf{S}$  in (21). According to Table 1, given sufficiently large  $N$  and  $K$ , the minimum achievable  $\eta$  for successful decoding is  $\eta = 1.23$  when  $d = 3$ , and hence the minimum measurement cost is  $2.46K$  if the regular ensemble is used in capturing the measurements. In order to achieve a measurement cost of  $M = 2K$  asymptotically, bipartite graphs with *irregular* left degrees need to be considered such that the redundancy parameter  $\eta$  can be made arbitrarily close to one  $\eta \rightarrow 1$ . Hence, for the noiseless setting particularly, we construct the coding matrix  $\mathbf{H}$  using an irregular graph ensemble rather than the regular graph ensemble  $\mathcal{G}_{\text{reg}}^N(R, d)$ .

We consider the irregular graph ensemble denoted by  $\mathcal{G}_{\text{irreg}}^N(R, D)$ , where each left node has irregular left degrees  $j = 2, \dots, D+1$ , where  $D+1$  is the maximum left degree. To describe the construction of the irregular graph ensemble, we use the left degree sequence  $\{\lambda_j\}_{j=2}^{D+1}$ , where  $\lambda_j$  is the fraction of edges<sup>15</sup> of degree  $j$  on the left<sup>16</sup>. For instance, the left degree sequence for the regular ensemble  $\mathcal{G}_{\text{reg}}^N(R, d)$  is  $\lambda_j = 1$  for  $j = d$  and but 0 if  $j \neq d$ .

**Definition 3** (Irregular Graph Ensemble  $\mathcal{G}_{\text{irreg}}^N(R, D)$  for Noiseless Recovery). *Given  $N$  left nodes and  $R = (1+\epsilon)K$  right nodes for an arbitrary  $\epsilon > 0$ , the edge set in the irregular graph ensemble  $\mathcal{G}_{\text{irreg}}^N(R, D)$  is characterized by the degree sequence*

$$\lambda_j = \frac{1}{H(D)(j-1)}, \quad j = 2, \dots, D+1 \quad (22)$$

where  $D > 1/\epsilon$  and  $H(D) = \sum_{j=1}^D 1/j$  is chosen such that  $\sum_{j \geq 2} \lambda_j = 1$ .

## 6.2 Oracle-based Peeling Decoder using the Irregular Ensemble $\mathcal{G}_{\text{irreg}}^N(R, D)$

Based on the peeling decoder analysis in Section 5.2, it can be easily shown that the concentration analysis and graph expansion property carry over to the irregular graph ensemble. Hence, we focus on the density evolution for the oracle-based peeling decoder over irregular ensemble.

To study the probability  $p_i$  of an edge being present in the pruned graph from the irregular ensemble after  $i$  iterations, we need to first understand the right edge degree distributions  $\rho_j$  of the graph. Using the degree sequence  $\lambda_j$  of the irregular graph ensemble  $\mathcal{G}_{\text{irreg}}^N(R, D)$  in Definition 3, it can be shown that the right degree sequence  $\rho_j$  follows a Poisson distribution similar to (11)

$$\rho_j \approx \frac{(\bar{d}/(1+\epsilon))^{j-1} e^{-\bar{d}/(1+\epsilon)}}{(j-1)!},$$

where we have used  $R = (1+\epsilon)K$  and  $\bar{d}$  is the average left degree of the irregular graph ensemble

$$\bar{d} = \frac{1}{\sum_{j=2}^{D+1} \lambda_j/j} = H(D) \left(1 + \frac{1}{D}\right). \quad (23)$$

<sup>15</sup>The graph is specified in terms of fractions of edges of each degree due to its notational convenience later on.

<sup>16</sup>An edge of degree  $j$  on the left (right) is an edge connecting to a left (right) node with degree  $j$ .

Using the left and right degree sequence  $(\lambda_j, \rho_j)$ , we can readily obtain the left and right degree generating polynomials  $\lambda(x) = \sum_{d=1}^{\infty} \lambda_j x^{j-1}$  and  $\rho(x) = \sum_{j=1}^{\infty} \rho_j x^{j-1}$

$$\lambda(x) = \frac{1}{H(D)} \sum_{j=2}^{D+1} \frac{1}{(j-1)} x^{j-1}, \quad \rho(x) = e^{-\frac{\bar{d}}{1+\epsilon}(1-x)}.$$

As a result, the associated density evolution equation can be written using the degree generating polynomials similar to that in (14)

$$p_i = f(p_{i-1}) = \lambda(1 - \rho(1 - p_{i-1})), \quad i = 1, 2, 3, \dots \quad (24)$$

The density evolution analysis suggests that if the fraction  $p_i$  in (24) can be made arbitrarily small if the density evolution recursion is contracting

$$\lambda(1 - \rho(1 - x)) < x, \quad \forall x \in [0, 1]. \quad (25)$$

Examples of this density evolution using different values of  $D$  and  $\epsilon$  are given in Fig. 8. Clearly, when  $\epsilon = 0.1$ , the density evolution equation becomes a contraction mapping when  $D = 100$  but not when  $D = 10$ . Now we study how to choose  $D$  for any given  $\epsilon > 0$ . Since  $\lambda(x)$  is a non-decreasing function, we can apply  $x = \lambda^{-1}(p_{i-1})$  on both sides of (25), then the contraction condition is equivalent to

$$\rho(1 - \lambda(x)) > 1 - x, \quad \forall x \in [0, 1]. \quad (26)$$

By substituting the right generating polynomial  $\rho(x)$  into the above recursion, we have

$$\rho(1 - \lambda(x)) = e^{-\frac{\bar{d}}{(1+\epsilon)}\lambda(x)}. \quad (27)$$

To simplify our expressions, we further bound  $\lambda(x)$  for the irregular graph ensemble  $\mathcal{G}_{\text{irreg}}^N(R, D)$  as  $\lambda(x) > -\frac{1}{H(D)} \log(1 - x)$ . This is because  $\lambda(x)$  is a  $D$ -term approximation of the Taylor expansion for  $\log(1 - x)$ , scaled by the normalization constant  $H(D)$ . By substituting this bound into (27), we have

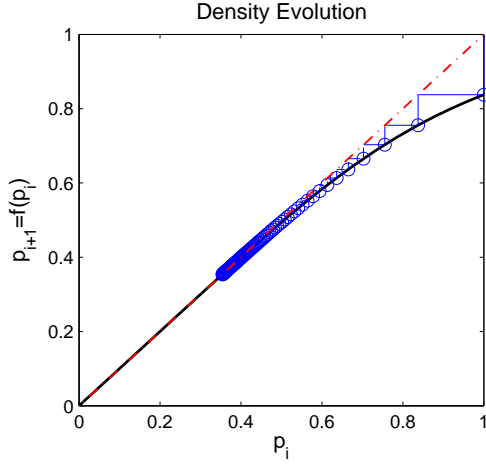
$$\rho(1 - \lambda(x)) > e^{\frac{\bar{d}}{(1+\epsilon)} \frac{1}{H(D)} \log(1-x)} = (1 - x)^{\frac{\bar{d}}{(1+\epsilon)H(D)}}.$$

It can be seen that the right hand side is no less than  $1 - x$  as long as  $H(D) \geq \frac{\bar{d}}{(1+\epsilon)}$ . Substituting the average degree  $\bar{d}$  from (23) back to this condition, then for any  $\epsilon > 0$ , we can choose  $D > 1/\epsilon$  as in Definition 3 to render the recursion a contracting mapping.

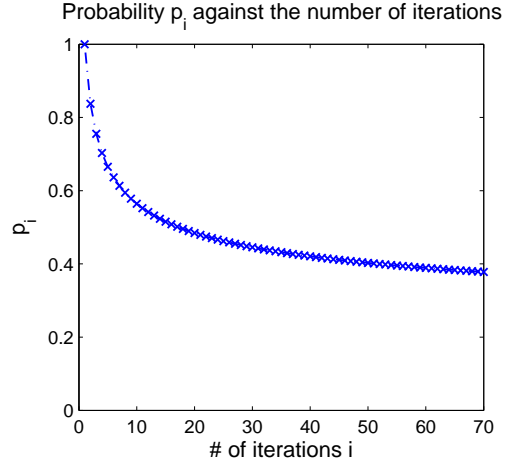
Finally, together with the concentration analysis and graph expansion properties of the irregular graphs, the oracle-based peeling decoder successfully decodes all the edges in the graph with probability at least  $1 - O(1/K)$ .

### 6.3 Getting Rid of the ‘Oracle’

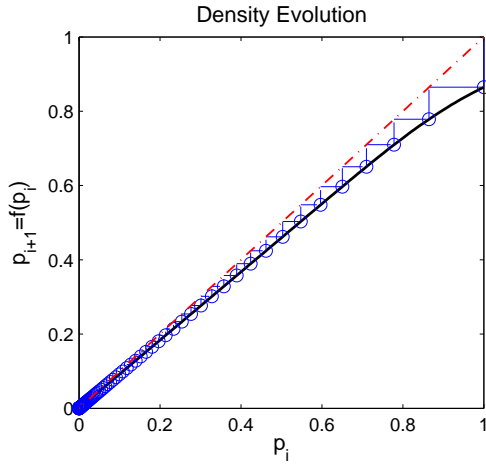
We have already shown that the oracle-based peeling decoder using the irregular graph successfully recovers all  $K$  unknown coefficients with high probability. In this section we discuss how to get rid of the oracle. We have briefly shown in Section 3.3 how to obtain the oracle information in the noiseless setting. In the following, we restate the procedures more formally to be self-contained. Using the two measurements in each bin  $\mathbf{y}_r = [y_r[0], y_r[1]]^T$  for  $r = 1, \dots, R$ , we perform the following tests to reliably identify the single-ton bins and obtain the correct index-value pair for any single-ton:



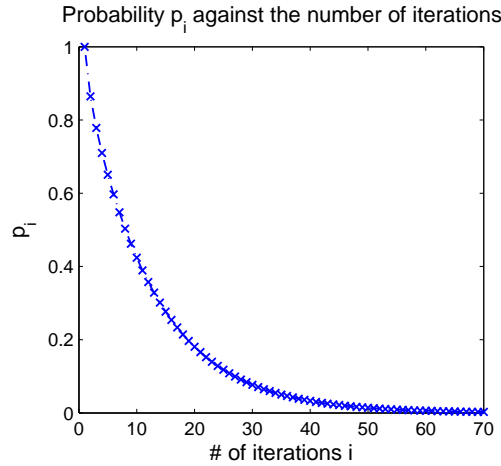
(a)  $\epsilon = 0.1$  and  $D = 10$



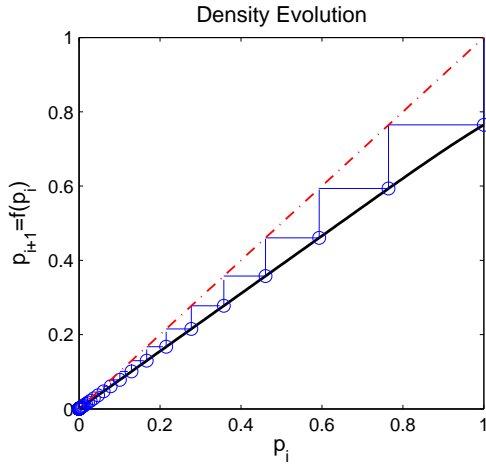
(b)  $\epsilon = 0.1$  and  $D = 10$



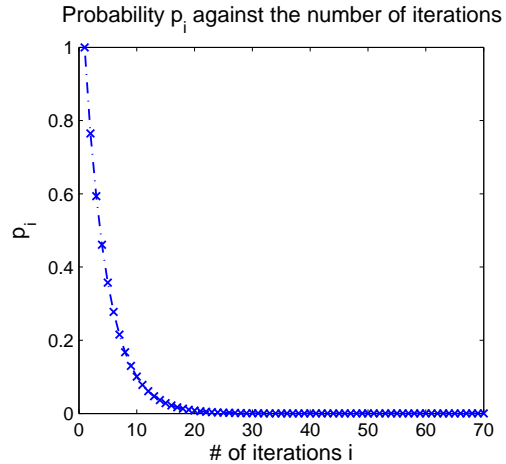
(c)  $\epsilon = 0.1$  and  $D = 100$



(d)  $\epsilon = 0.1$  and  $D = 100$



(e)  $\epsilon = 0.3$  and  $D = 100$



(f)  $\epsilon = 0.3$  and  $D = 100$

Figure 8: The density evolution  $f(p_i)$  and the probability  $p_i$  at each iteration  $i$ , where we have shown cases with  $\epsilon = 0.1$  and  $D = 10$  and  $D = 100$ , as well as the case with  $\epsilon = 0.3$  and  $D = 100$ . In the density evolution figures (a)-(c)-(e), the red line is the line  $p_{i+1} = p_i$  while the black line is the density evolution  $f(p_i)$  against  $p_i$ . The blue circles that “zig-zag” between the red line and the black line are the specific  $p_i$ ’s at each peeling iteration. It can be seen from (a) and (c) that when  $\epsilon$  is small (i.e.  $\epsilon = 0.1$ ), the density evolution requires a large maximum left degree  $D$  to reach density 0. On the other hand, when  $\epsilon$  is large (i.e.  $\epsilon = 0.3$ ), the density  $p_i$  reaches 0 very quickly in (e) with the same maximum left degree  $D = 100$ . The values of  $p_i$  marked by the blue circles in (a)-(c)-(e) are further plotted against the peeling iterations  $i$  in (b)-(d)-(f), where in the case with  $\epsilon = 0.3$  and  $D = 100$  the density  $p_i$  approaches 0 after less than 20 iterations.

- **Zero-ton Test:** since there is no noise, it is clear that the bin is a zero-ton if  $\|\mathbf{y}_r\|^2 = 0$ .
- **Multi-ton Test:** The measurement bin is a multi-ton as long as  $|y_r[1]| \neq |y_r[0]|$  and/or

$$\angle \frac{y_r[1]}{y_r[0]} \neq 0 \pmod{2\pi/N}.$$

The multi-ton test fails when the relative phase is a multiple of  $2\pi/N$ , which corresponds to the following condition according to the measurement model in (8)

$$\frac{y_r[1]}{y_r[0]} = \frac{\sum_{n \in [N]} H_{r,n} x[n] G_n e^{i \frac{2\pi n \ell}{N}}}{\sum_{n \in [N]} H_{r,n} x[n] G_n} = e^{i \frac{2\pi \ell}{N}}, \quad \text{for some } \ell \in [N] \quad (28)$$

where  $H_{r,n}$  is the  $(r, n)$ -th entry in the coding matrix  $\mathbf{H}$ . Clearly, this event is measure zero under the continuous distribution of  $G_n$  for  $n \in [N]$ .

- **Single-ton Test:** After the zero-ton and multi-ton tests, the measurement bin is detected as a single-ton if  $|y_r[1]| = |y_r[0]|$  and

$$\angle \frac{y_r[1]}{y_r[0]} = 0 \pmod{2\pi/N}$$

where

$$\hat{k}_r = \frac{N}{2\pi} \angle \frac{y_r[1]}{y_r[0]}, \quad \hat{x}[\hat{k}_r] = y_r[0]. \quad (29)$$

This gives us the index-value pair of the single-ton for peeling.

## 7 Noisy Recovery

### 7.1 Measurement Construction

Since we propose two designs for noisy recovery with different run-time, we first provide the constructions of the *bin detection matrix*  $\mathbf{S}$  for the two designs separately.

**Definition 4** (Random Matrix for Near-Linear Run-time). *The  $P \times N$  bin detection matrix  $\mathbf{S}$  with  $P = O(\log N)$  is constructed as a random matrix with zero-mean unit-variance i.i.d. sub-gaussian entries.*

**Definition 5** (Partially Random DFT Matrix for Sub-linear Run-time). *Let  $P = CQ$  for some  $C = O(\log_2 N)$  and  $Q = O(\log^{0.3} N)$  where  $N$  is not a multiple<sup>17</sup> of 2. The  $P \times N$  bin detection matrix  $\mathbf{S}$  consists of  $C$  sub-matrices of size  $Q \times N$*

$$\mathbf{S} = [\mathbf{S}_0^\dagger \quad \mathbf{S}_1^\dagger \quad \cdots \quad \mathbf{S}_{C-1}^\dagger]^\dagger. \quad (30)$$

---

<sup>17</sup>The base element 2 is chosen for simplicity, and can be generalized to any integer.

Each sub-matrix  $\mathbf{S}_c$  consists of  $Q$  consecutive  $2^c$ -dyadically spaced rows<sup>18</sup> from the  $N \times N$  DFT matrix, starting from a random row  $\ell_c \in [N]$

$$\mathbf{S}_c = \begin{bmatrix} 1 & W^{\ell_c} & \dots & W^{n\ell_c} & \dots \\ 1 & W^{\ell_c+2^c} & \dots & W^{n(\ell_c+2^c)} & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & W^{\ell_c+(Q-1)2^c} & \dots & W^{n(\ell_c+(Q-1)2^c)} & \dots \end{bmatrix},$$

where  $W = e^{i\frac{2\pi}{N}}$  is the  $N$ -th root of unity.

Note that when it comes to the coding matrix  $\mathbf{H}$ , we can certainly use the irregular graph ensemble as in the noiseless case because it gives sharper measurement bound. However, since we are providing order-wise results for the measurement costs for noisy results, we consider the regular graph ensemble  $\mathcal{G}_{\text{reg}}^N(R, d)$  instead because of its simplicity in constructions.

## 7.2 Robust Bin Detection

Unlike the noiseless setting, the ratio tests for identifying single-tons no longer work because of the measurement noise. In this general setting, the role of the *bin detection matrix*  $\mathbf{S}$  is critical in distinguishing the bin hypotheses (see Proposition 1) from one another. Although the constructions for the *bin detection matrix*  $\mathbf{S}$  are different in Definition 4 and 5, which lead to different robust bin detection schemes, we show that the peeling decoder using both designs without an oracle achieves the same performance as that endowed with an oracle.

**Proposition 1.** *Given the bin detection matrix  $\mathbf{S}$  in Definition 4 and 5, the failure probability  $\mathbb{P}_F$  of the proposed peeling decoder without an oracle is bounded by  $O(1/K)$ .*

*Proof.* We refer the readers to the Appendix in our full report available at [62].  $\square$

In the following, we present the robust bin detection schemes for our designs in the noisy setting. Since the procedures are the same for any measurement bin at any iteration, we drop the bin index  $r$  in (8) and use the italic font  $\mathbf{y}$  to denote a generic bin measurement  $\mathbf{y}_r$  using the following model

$$\mathbf{y} = \mathbf{S}\mathbf{z} + \mathbf{w} \quad (31)$$

for some sparse vector  $\mathbf{z}$ . Clearly, the sparse vector equals  $\mathbf{z} = \text{diag}[\mathbf{h}]\mathbf{x}$  in the first iteration, where  $\mathbf{h}^T = [H_1, \dots, H_N]$  refers to some row of the coding matrix  $\mathbf{H}$  with the row index  $r$  dropped<sup>19</sup>, but as the peeling iterations proceed, the non-zero coefficients in  $\mathbf{z}$  will be peeled off and potentially left with a 1-sparse coefficient. Therefore, at each iteration, we perform the following zero-ton and single-ton tests to verify if  $\mathbf{z}$  has become a 1-sparse signal and identify its index-value pair.

For zero-ton bins, it is expected that the energy  $\|\mathbf{y}\|^2$  to be small relative to the energy of a single-ton. Therefore, this idea is used to eliminate zero-tons:

$$\hat{\mathcal{H}} = \mathcal{H}_Z, \quad \text{if } \frac{1}{P} \|\mathbf{y}\|^2 \leq (1 + \gamma)\sigma^2 \quad (32)$$

<sup>18</sup>The reason for choosing the dyadic spacing is because  $N$  is not a multiple of 2. Since  $N$  is not a multiple of 2, the dyadic nature does not lead to repetitive choices of the DFT rows due to the periodic wrap-around of the DFT matrix. Therefore, in general, the spacing can be chosen arbitrarily to be the power of any constant  $N_*$  as long as the signal length does not have  $N_*$  as a multiplicative factor. This only changes the constants in our big-Oh statements, but not the scaling.

<sup>19</sup>The index is dropped for the analysis of an arbitrary bin (i.e., arbitrary row of the coding matrix). The transpose operator is used to maintain  $\mathbf{h}$  as a column vector for notational convenience in our analysis.



for some  $\gamma \in (0, 1)$ . After ruling out zero-tons, the goal is to identify the index-value pairs  $(k, x[k])$  of single-tons, and peel them off from multi-ton bins. The single-ton test consists of

- the **single-ton search**, which estimates the index-value pair  $(\hat{k}, \hat{x}[\hat{k}])$  assuming that the underlying bin is a single-ton, denoted by  $\psi : \mathbf{y} \rightarrow (\hat{k}, \hat{x}[\hat{k}])$ ,
- the **single-ton verification**, which confirms whether the bin is in fact a single-ton with the estimated index-value pair via the residual test

$$\hat{\mathcal{H}} = \mathcal{H}_s(\hat{k}, \hat{x}[\hat{k}]) \quad \text{if } \frac{1}{P} \left\| \mathbf{y} - \hat{x}[\hat{k}] \mathbf{s}_{\hat{k}} \right\|^2 \leq (1 + \gamma) \sigma^2. \quad (33)$$

All the tests above except for the *single-ton search* are identical for both of our noisy designs, which are summarized in Algorithm 1.

---

**Algorithm 1** Robust Bin Detector  $\varphi : \mathbf{y} \rightarrow \hat{\mathcal{H}}$

---

Input : Observations  $\mathbf{y}$  for bin  $j = 1, \dots, \eta K$ .  
Set : Parameter  $\gamma \in (0, 1)$ .  
Output : Bin type  $\hat{\mathcal{H}}$  and index-value pair  $(\hat{k}, \hat{x}[\hat{k}])$   
**if**  $\|\mathbf{y}\|^2 / P \leq (1 + \gamma) \sigma^2$  **then**  
    **return**  $\hat{\mathcal{H}} = \mathcal{H}_Z$      % Zero-ton Test  
**else**  
     $\psi : \mathbf{y} \rightarrow (\hat{k}, \hat{x}[\hat{k}])$ , % Single-ton Search  
    **if**  $\left\| \mathbf{y} - \hat{x}[\hat{k}] \mathbf{s}_{\hat{k}} \right\|^2 / P \leq (1 + \gamma) \sigma^2$  **then**  
        **return**  $\hat{\mathcal{H}} = \mathcal{H}_s(\hat{k}, \hat{x}[\hat{k}])$  % Single-ton Verification  
    **else**  
        **return**  $\hat{\mathcal{H}} = \mathcal{H}_M$   
    **end if**  
**end if**

---

In the following, we discuss the single-ton search methods using different constructions of  $\mathbf{S}$  that lead to different operating points  $(M, T)$ .

### 7.3 Single-ton Search $\psi : \mathbf{y} \rightarrow (\hat{k}, \hat{x}[\hat{k}])$ in Robust Bin Detectors

#### 7.3.1 Near-linear Time Single-ton Search

In this setting, the *bin detection matrix* is chosen as Definition 4. A generalized likelihood test is used in the *single-ton search*. For each possible coefficient index  $k$ , we obtain the maximum likelihood (ML) of the coefficient as:

$$a_k = \frac{\mathbf{s}_k^\dagger \mathbf{y}}{\|\mathbf{s}_k\|^2}. \quad (34)$$

Substituting the estimate of the coefficient  $a_k$  into the likelihood of the particular single-ton hypothesis in Proposition 1, we choose the index  $k$  that minimizes the residual energy:

$$\hat{k} = \arg \min_{k \in \text{supp}(\mathbf{h})} \left\| \mathbf{y} - a_k \mathbf{s}_k \right\|^2. \quad (35)$$

The search is over  $k \in \text{supp}(\mathbf{h})$  because the support of  $\mathbf{z}$  never goes out of the support of the coding pattern, which is known a priori. With the estimated index  $\hat{k}$ , the coefficient is obtained by aligning it to the closest alphabet symbol in  $\mathcal{X}$

$$\hat{x}[\hat{k}] = \min_{x \in \mathcal{X}} \|a_{\hat{k}} - x\|^2. \quad (36)$$

Solving the above minimizations requires an exhaustive search over all elements in  $\mathcal{X}$  and all indices  $k \in \text{supp}(\mathbf{h})$ , whose complexity grows at least linearly with the signal dimension  $N$ . In the following, we introduce how to perform single-ton searches in sub-linear time using a different bin detection matrix.

### 7.3.2 Sub-linear Time Single-ton Search

Here we exploit a similar approach to that for the noiseless case in Theorem 1. In particular, we perform the single-ton search of the index-value pair  $(k, x[k])$  by using the Fourier structure of the *bin detection matrix*  $\mathbf{S}$  in Definition 5. In particular, estimating the index  $k$  can be viewed as the estimation of a discrete frequency of a complex sinusoid given by each column of the DFT matrix

$$\omega = \frac{2\pi k}{N} \quad (37)$$

using spectral estimation methods that have been extensively studied in signal processing [61]. This is a more systematic version of the ratio test for the noiseless case.

To be more specific, the measurement  $\mathbf{y}_c = \mathbf{S}_c \mathbf{z} + \mathbf{w}_c$  associated with each sub-matrix  $\mathbf{S}_c$  in the *bin detection matrix*  $\mathbf{S}$  in Definition 5 is called a “cluster”, which correspond to the periodic samples that are taken with a dyadic spacing  $2^c$ . Without loss of generality, let us consider the  $c$ -th block in the *bin detection matrix*  $\mathbf{S}$  in Definition 5, which consists of  $Q$  consecutive rows (dyadically spaced) from the DFT matrix. If the underlying bin is in fact a single-ton with some index-value pair  $(k, x[k])$ , the measurements  $\mathbf{y}_c$  correspond to the consecutive samples of a complex sinusoid

$$y_c[m] = x[k]e^{i\frac{2\pi k \times 2^c}{N}m} + w_c[m], \quad m = 0, \dots, Q-1 \quad (38)$$

with some frequency  $\omega_c = 2\pi k/N \times 2^c$  and complex amplitude  $x[k]$ . Equivalently, the samples obtained by this structured DFT matrix can be visualized as clusters of samples from a complex sinusoid in Fig. 9.

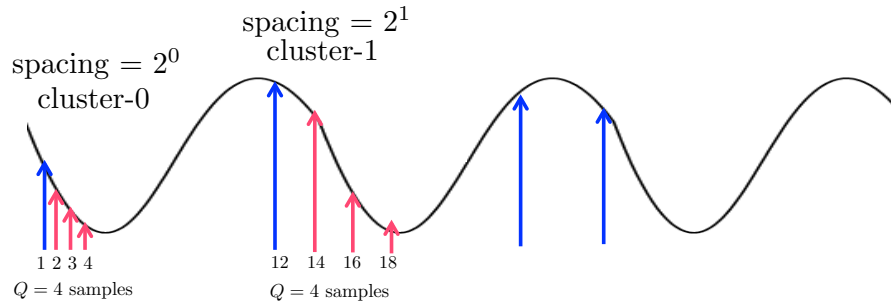


Figure 9: Example of the real/imaginary part of the observation obtained by the structured random DFT matrix in Definition 5, with  $C = 4$  clusters and  $Q = 4$  samples in each cluster.

A reliable way to estimate the frequency  $\omega_c$  and coefficient is through an ML estimator, which however leads to high complexity. Therefore, we leverage the unbiased and efficient linear estimator developed in [61] and propose a successive spectral estimation scheme for our single-ton search.

**Lemma 4** (Frequency Estimation [61]). *Given the complex sinusoid in (38) with  $w[m] \sim \mathcal{N}(0, \sigma^2)$  and a sufficiently large signal-to-noise ratio<sup>20</sup>  $|x[k]|^2/\sigma^2$ , the following estimator*

$$\hat{\omega}_c = \sum_{m=0}^{Q-2} r_m \angle y_c^*[m] y_c[m+1] \quad (39)$$

with weights  $r_m := \frac{3Q/2}{Q^2-1} \left(1 - \left[\frac{m-(Q/2-1)}{Q/2}\right]^2\right)$  has an estimation error  $\Delta_c = \hat{\omega}_c - \omega_c \sim \mathcal{N}\left(0, \frac{6\sigma^2/|x[k]|^2}{Q(Q^2-1)}\right)$ .

**Corollary 2.** *The estimate  $\hat{\omega}_c$  from Lemma 4 satisfies*

$$\Pr\left(|\Delta_c| > \frac{\pi}{2}\right) \leq 4 \exp\left(-\frac{A_{\min}^2}{4\sigma^2} Q^3\right). \quad (40)$$

*Proof.* We refer the readers to the Appendix in our full report available at [62].  $\square$

Nonetheless, the index  $k$  can be estimated without ambiguity from the first cluster  $\omega_0 = 2\pi k/N$ , since there are  $2^c$  possible choices of  $k$  that results in the same frequency  $\omega_c = 2\pi k/N \times 2^c$ . Therefore, the linear estimator in Lemma 4 would require using cluster-0 and require the resulting estimation error to be sufficiently small  $\Delta_0 < 1/N$  with high probability, such that  $\omega_0$  can be obtained correctly within  $[-\pi/N, \pi/N]$ . From the distribution of the estimation error  $\Delta_0$ , this implies a measurement scaling of  $Q = O(N^{1/3})$ , which is polynomial in  $N$  and undesirable for fast implementation. However, if we further exploit the multiple clusters and analyze the ambiguous pattern of each  $\omega_c$ , we can improve the scaling dramatically by performing such estimation recursively over  $C$  clusters of  $Q = O(\log^{1/3} N)$  measurements. This is what constitutes the sub-linear time single-ton search via successive estimation over multiple clusters, which is explained below.

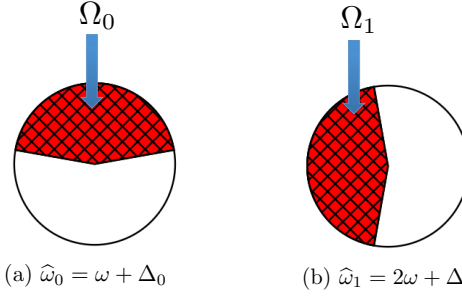
### Step 1: Cluster Estimation

According to Lemma 4, the measurement  $\mathbf{y}_c$  in the  $c$ -th cluster provides an estimate  $\hat{\omega}_c$  of  $2^c\omega$  modulo  $2\pi$ . Let  $\Omega_0 = (\omega_0 - \pi/2, \omega_0 + \pi/2)$  be the set of frequencies around the estimate  $\omega_0$  obtained by processing cluster-0, which we call the *certainty region*. It is known from Corollary 2 that with some probability  $\lim_{Q \rightarrow \infty} \Pr(\omega \notin \Omega_0) \rightarrow 0$ , the estimate falls outside the certainty region. Similarly, for the second cluster, with some other probability  $\lim_{Q \rightarrow \infty} \Pr(2\omega \notin \Omega_1) \rightarrow 0$ , the estimate falls outside the certainty region  $\Omega_1 = (\omega_1 - \pi/2, \omega_1 + \pi/2)$ . The certainty regions are illustrated for the first two clusters in Fig. 10a, whereas the same holds for other clusters over  $c \in [C]$ .

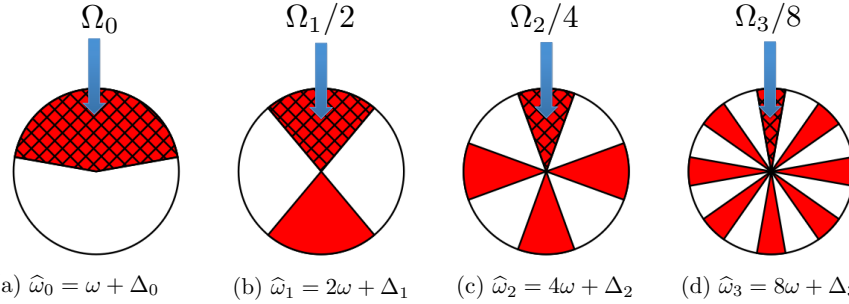
### Step 2: Unwrapping

We define the *unwrap* procedure to infer  $\omega$  from each estimate of  $2^c\omega$  in the  $c$ -th cluster. Since  $\omega = 2\pi k/N$  for some  $k \in [N]$ , the factor of  $2^c$  essentially maps  $2^c$  frequencies to the same frequency  $\omega_c = 2^c\omega$  (modulo  $2\pi$ ), one of which is the ground truth  $\omega$ . Therefore, the certainty region  $\Omega_c$  can be unwrapped into  $2^c$  much smaller slices of *unwrapped certainty regions* of width  $\pi/(2^c)$  over the unit circle. We denote the set of all *unwrapped certainty regions* as  $\Omega_c/2^c$ . Note that the certainty region  $\Omega_c$  and the unwrapped certainty region  $\Omega_c/2^c$  have the same cardinality  $|\Omega_c| = |\Omega_c/2^c| = \pi$ , but their occupancies on the unit circle is different, as shown in Fig. 10b.

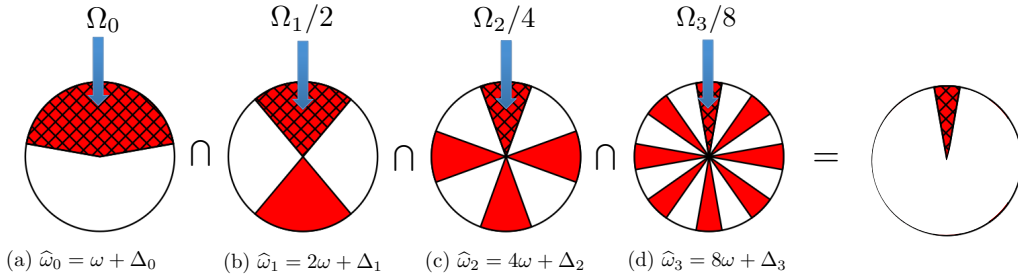
<sup>20</sup>The author of [61] has empirically validated such signal-to-noise ratio requirement to be  $5 \sim 7$ dB.



(a) Example of cluster estimations over the first two clusters, where the shape of the red pie indicates the uncertainty region  $\Delta_i$  for the cluster estimate  $\hat{\omega}_i$  of the frequency  $2^i\omega$ .



(b) Example of frequency unwrapping over multiple clusters, from left (cluster 0) to right (cluster 3). Cluster-1 provides the unwrapped uncertainty region consisting of 2 slices due to the frequency estimate of  $2\omega$ , cluster-2 provides unwrapped certainty region consisting of 4 slices due to the frequency estimate of  $4\omega$ , cluster-3 provides unwrapped certainty region consisting of 8 slices due to the frequency estimate of  $8\omega$ .



(c) Example of estimate fusion over multiple clusters, from left (cluster 0) to right (cluster 3), which leads to the intersection of the reduced slice of pie on the right.

### Step 3: Estimate Fusion

Based on the unwrapped certainty regions over multiple clusters, the frequency estimate  $\hat{\omega} = 2\pi\hat{k}/N$  is refined successively by intersecting the  $C$  unwrapped certainty regions

$$\Omega := \bigcap_{c=0}^{C-1} \Omega_c/2^c. \quad (41)$$

Since  $|\Omega_0| = \pi$  and  $N$  is sufficiently large, the intersection of the first two clusters leads to a halved uncertainty interval  $|\Omega_0 \cap \Omega_1/2| \approx \pi/2$  and thus intuitively, the size of the set  $\Omega$  shrinks down exponentially with the number of clusters  $C$ , as shown in Fig. 10c.

#### Step 4: Final Estimate

As mentioned in the estimate fusion step, the successive intersection over multiple clusters sharpens the certainty set  $\Omega$  in terms of its cardinality, which shrinks exponentially with respect to the number of clusters. Clearly, as long as we have  $C = O(\log N)$  clusters, the cardinality of the final certainty set  $\Omega$  can be made as small as  $|\Omega| \leq \pi/N$  such that it only contains  $O(1)$  possible discrete frequencies  $\omega$  with high probability. Furthermore, with some probability  $\lim_{Q \rightarrow \infty} \Pr(\omega \notin \Omega_c/2^c) \rightarrow 0$  that vanishes exponentially  $Q$  according to Corollary 2, the estimate falls outside the unwrapped certainty region, the probability of the estimate lying outside at least one of the unwrapped certainty regions can be upper bounded by a union bound over  $C$  clusters, which remains vanishing to 0 asymptotically in  $Q$  irrespective of  $C$ . Therefore, as long as each cluster has  $Q = O(\log^{1/3} N)$  measurements, the ground truth will be in the final certainty region  $\Omega$  with probability at least  $1 - O(1/N)$ . As a result, we can easily find an estimate  $\hat{k}$  with  $2\pi\hat{k}/N \in \Omega$  over these few possible candidates via (35) in Section 7.3.1.

## 8 Numerical Experiments

In this section, we provide the empirical performance of our design in the noiseless and noisy settings. Each data point in the simulation is generated by averaging over 200 experiments, where the signals  $\mathbf{x}$  are generated once and kept fixed for all the subsequent experiments. In particular, the support of  $\mathbf{x}$  are generated uniformly random from  $[N]$  with values chosen from the set  $\{\pm 1\}$ . In the presence of noise, the signal-to-noise ratio (SNR) is defined as

$$\text{SNR} = \frac{\mathbb{E}[\|\mathbf{B}\mathbf{x}\|^2]}{\mathbb{E}[\|\mathbf{w}\|^2]} = \frac{\|\mathbf{x}\|^2}{\sigma^2} \frac{\bar{d}}{R} \quad (42)$$

where  $\bar{d}$  is the average left node degree of the bipartite graph,  $R$  is the number of right nodes in the graph, and the expectation is taken with respect to the noise, *random bipartite graph* and *bin detection matrix*. Then in noisy settings, we generate i.i.d. circularly complex Gaussian noise with variance  $\sigma^2$  according to the specified SNR for each  $\mathbf{x}$ .

#### *Density Evolution using the Irregular Ensemble and the Regular Ensemble*

In this case, we examine the density evolution results using our noiseless recovery algorithm in Section 6. We generate a random  $\mathbf{x}$  with  $K = 500$  and  $N = 10^4$  for all the experiments. To understand the effects of using different graphs on the density evolution, we numerically verify the density evolution threshold  $R/K$  using the regular and irregular graph ensemble by tracing the probability of success  $1 - \mathbb{P}_F$  against the redundancy parameter  $\eta = R/K$ . On one hand, for the regular graph  $\mathcal{G}_{\text{reg}}^N(R, d)$  we fix  $d = 3$  and vary the redundancy parameter  $\eta = R/K$  from 1 to 1.5. It can be seen that the threshold for  $R/K = \eta$  empirically matches with the density evolution analysis for regular graphs in Section 5.2, where the algorithm succeeds with some probability from  $\eta = 1.2$  and reaches probability one after  $\eta = 1.3$ . On the other hand, for the irregular graph ensemble  $\mathcal{G}_{\text{irreg}}^N(R, D)$ , we fix the maximum left degree  $D = 100$  and vary the redundancy parameter  $R/K = (1 + \epsilon)$ . It can be seen in Fig. 10 that the algorithm succeeds with some probability from  $\epsilon = 0.1$ , which matches with our density evolution analysis for the irregular ensemble  $\epsilon > 1/D$  in Section 6. The probability of success using the irregular graph ensemble approaches one more slowly because it requires much larger  $K$  and  $N$  to reach probability one at  $\epsilon = 0.1$ , which is not presented here.

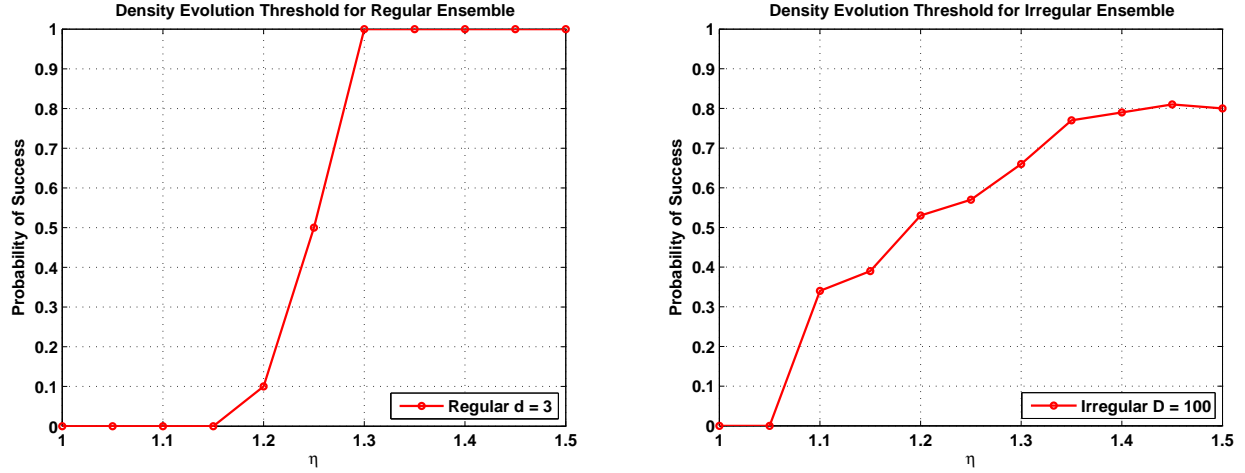


Figure 10: Probability of success  $1 - \mathbb{P}_F$  for signals of length  $N = 0.1$  million using the regular and irregular ensemble. The regular graph ensemble has a fixed left regular degree  $d = 3$  and the probability of success is plotted against the redundancy parameter  $\eta = R/K$ , whereas the irregular graph ensemble has a fixed maximum left degree  $D = 100$  and the probability of success is plotted against the redundancy parameter  $R/K = 1 + \epsilon$ . Note that in both plots we keep our notations consistent by specifying  $\eta$  instead of  $\epsilon$  for the irregular graph, but the value of  $\epsilon$  can be easily obtained as  $\epsilon = \eta - 1$ .

#### Scalability of Measurement and Computational Costs for Noiseless Recovery

In this case, we examine the measurement cost and run-time of our noiseless recovery algorithm. The measurement matrix  $\mathbf{B}$  is constructed using the coding matrix  $\mathbf{H}$  from the irregular graph ensemble  $\mathcal{G}_{\text{irreg}}^N(R, D)$  by fixing  $R = 1.1K$  and  $D = 100$ . We show experiments with different sparsities where  $K = 200, 400$  and  $600$  and for each of the sparsity settings, we simulate our noiseless recovery algorithm for recovering sparse signals of dimension  $N = 10^4$  to  $N = 7 \times 10^4$ . It can be seen in Fig. 11 that the measurement and computational costs remain constant irrespective of the growth in  $N$ .

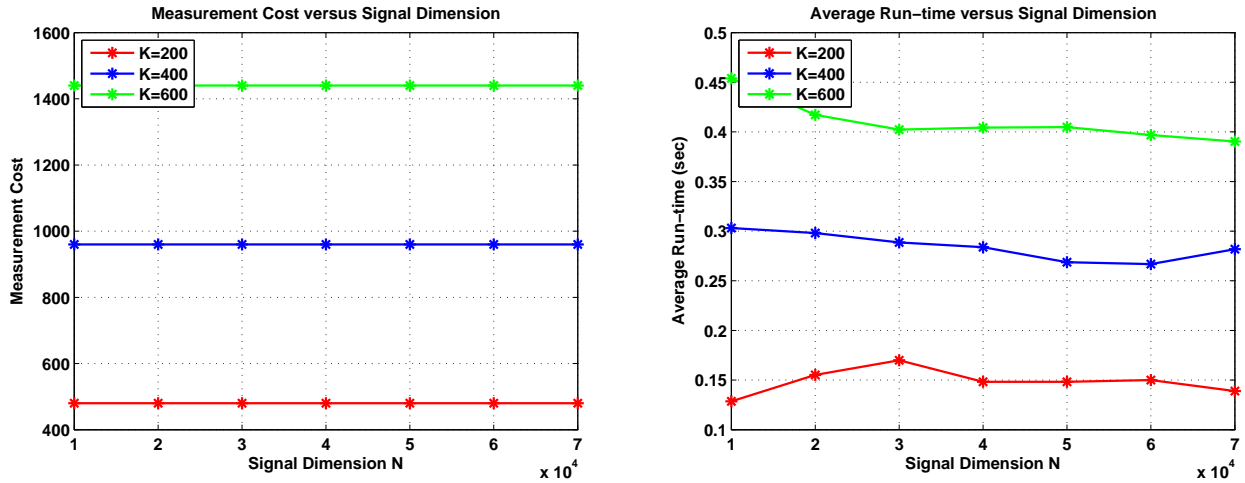


Figure 11: Measurement and computational costs as functions of the signal dimension  $N$  for noiseless recovery.

## Noise Robustness

In this section, we showcase our near-linear time noisy recovery scheme and sub-linear time noisy recovery scheme using the *bin detection matrix*  $\mathbf{S}$  from Definition 4 and Definition 5 respectively. In each experiment, 50-sparse signals  $\mathbf{x}$  (i.e.,  $K = 50$ ) with  $N = 10^5$  are generated. The measurement matrix  $\mathbf{B}$  is constructed from the coding matrix  $\mathbf{H}$  using the regular graph ensemble  $\mathcal{G}_{\text{reg}}^N(R, d)$  with a regular degree  $d = 3$  and  $R = 2K$ . To demonstrate the noise robustness, the probability of success is plotted against a range of SNR from 0dB to 16dB using  $M = 2KP$  measurements with  $P = 3 \log N$  for the near-linear time recovery according to Definition 4 and  $P = 3 \log^{1.3} N$  for the sub-linear time recovery according to Definition 5. It is seen in Fig. 12 that for a given measurement cost, there exists a threshold of SNR, above which our noisy recovery schemes succeed with probability 1. It is also observed that the thresholds increase gracefully when the measurement cost is reduced.

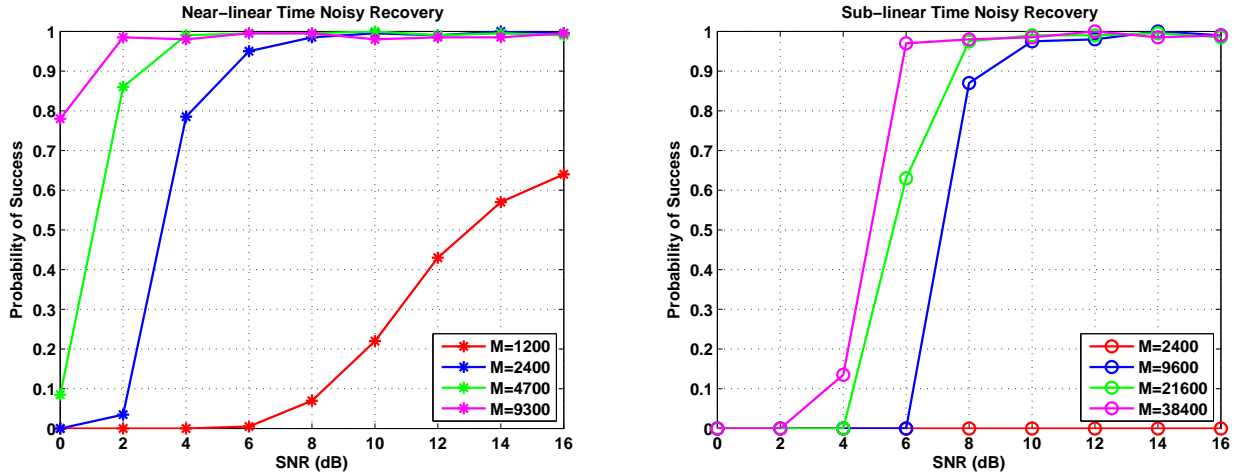


Figure 12: Probability of success against SNR for  $N = 0.1$  million and  $K = 50$ .

## Scalability of Measurement and Computational Costs for Noisy Recovery

In this section, we demonstrate how the measurement costs and algorithm run-time of our near-linear time noisy recovery scheme and sub-linear time noisy recovery scheme grow with respect to the problem dimension  $N$ . In each experiment, the sparsity of the  $K$ -sparse signals  $\mathbf{x}$  is chosen as  $K = N^\delta$  under different sparsity regimes  $\delta = 1/6, 1/3$  and  $1/2$ , while the ambient dimensions of the signals for each sparsity regime ranges from  $N = 10^2$  to  $N = 10^7 \approx 10$  million. The measurements are obtained under the SNR of 10dB. The coding matrix  $\mathbf{H}$  is constructed from the regular graph ensemble  $\mathcal{G}_{\text{reg}}^N(R, d)$  with a regular degree  $d = 3$  and  $R = 2K$ , and the *bin detection matrix*  $\mathbf{S}$  is chosen separately according to Definition 4 with  $P = 3 \log N$  and Definition 5 with  $P = 3 \log^{1.3} N$ . It can be seen in Fig. 13 that the while the measurement cost scales sub-linearly for the near-linear time noisy recovery scheme, the run-times for different sparse signals almost overlap with each other, which depends linearly on the signal dimension  $N$  as stated in Theorem 2. In contrast, our sub-linear time recovery scheme achieves sub-linear scaling on both measurement and computational costs, as stated in Theorem 3.

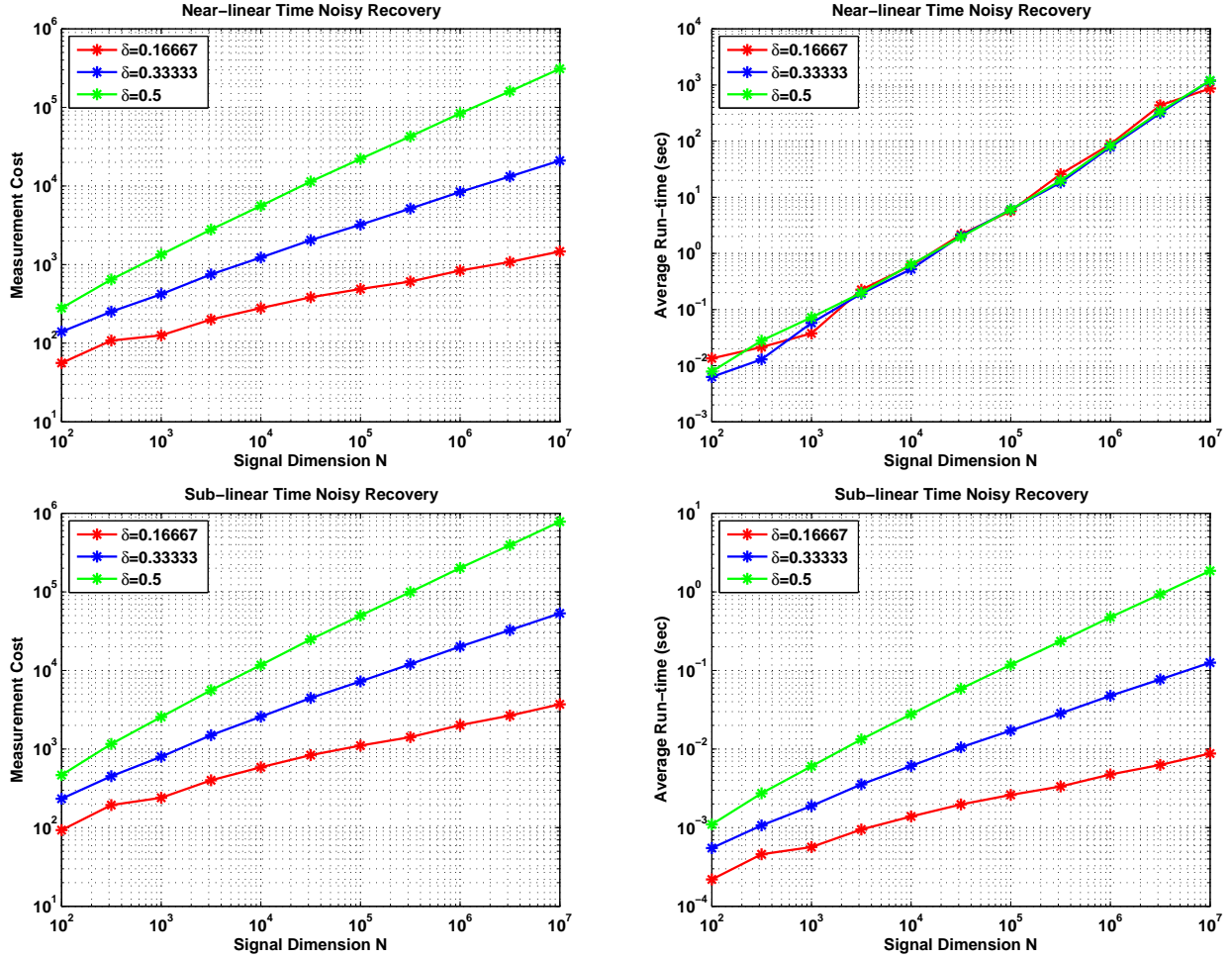


Figure 13: Measurement and computational costs as functions of the signal dimension  $N$  for noisy recovery. It can be seen that the measurement cost of the sub-linear time recovery is on par with the near-linear time recovery, but the run-time of our sub-linear time recovery scheme differs drastically from the near-linear time recovery scheme. For instance, when  $N = 10$  million and  $K = \sqrt{N}$  (the green curves on both plots), the measurement costs for both schemes are approximately  $10^6$  while the run-time is 1000 seconds for the near-linear time recovery scheme and less than 10 seconds for the sub-linear time recovery scheme.

## 9 Conclusions

In this paper, we addressed the support recovery problem for compressed sensing using sparse-graph codes. We proposed a compressed sensing design framework for sub-linear time support recovery, by introducing a new family of measurement matrices and fast recovery algorithms. We also provide simulation results to corroborate our theoretical findings.

## References

- [1] D. L. Donoho, “Compressed sensing,” *IEEE Trans. on Information Theory*, vol. 52, no. 4, pp. 1289–1306, 2006.



- [2] M. Lustig, D. Donoho, and J. M. Pauly, “Sparse mri: The application of compressed sensing for rapid mr imaging,” *Magnetic resonance in medicine*, vol. 58, no. 6, pp. 1182–1195, 2007.
- [3] E. J. Candes, Y. C. Eldar, T. Strohmer, and V. Voroninski, “Phase retrieval via matrix completion,” *SIAM Journal on Imaging Sciences*, vol. 6, no. 1, pp. 199–225, 2013.
- [4] M. Elad, *Sparse and redundant representations: from theory to applications in signal and image processing*. Springer, 2010.
- [5] A. Gilbert and P. Indyk, “Sparse recovery using sparse matrices.” Institute of Electrical and Electronics Engineers, 2010.
- [6] R. G. Baraniuk, “Compressive sensing,” *IEEE signal processing magazine*, vol. 24, no. 4, 2007.
- [7] E. J. Candès, Y. Plan *et al.*, “Near-ideal model selection by  $\ell_1$  minimization,” *The Annals of Statistics*, vol. 37, no. 5A, pp. 2145–2177, 2009.
- [8] D.-Z. Du and F. K. Hwang, *Combinatorial group testing and its applications*. World Scientific, 1993.
- [9] J.-J. Fuchs, “Recovery of exact sparse representations in the presence of bounded noise,” *IEEE Trans. on Information Theory*, vol. 51, no. 10, pp. 3601–3608, 2005.
- [10] E. Greenshtein *et al.*, “Best subset selection, persistence in high-dimensional statistical learning and optimization under  $\ell_1$  constraint,” *The Annals of Statistics*, vol. 34, no. 5, pp. 2367–2386, 2006.
- [11] S. Pawar and K. Ramchandran, “Computing a  $k$ -sparse  $n$ -length discrete fourier transform using at most  $4k$  samples and  $\mathcal{O}(k \log k)$  complexity,” in *Information Theory Proceedings (ISIT), 2013 IEEE International Symposium on*. IEEE, 2013, pp. 464–468.
- [12] R. Tibshirani, “Regression shrinkage and selection via the lasso,” *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 267–288, 1996.
- [13] T. Blumensath and M. E. Davies, “Iterative hard thresholding for compressed sensing,” *Applied and Computational Harmonic Analysis*, vol. 27, no. 3, pp. 265–274, 2009.
- [14] A. Beck and M. Teboulle, “A fast iterative shrinkage-thresholding algorithm for linear inverse problems,” *SIAM Journal on Imaging Sciences*, vol. 2, no. 1, pp. 183–202, 2009.
- [15] D. L. Donoho, A. Maleki, and A. Montanari, “Message-passing algorithms for compressed sensing,” *Proceedings of the National Academy of Sciences*, vol. 106, no. 45, pp. 18 914–18 919, 2009.
- [16] E. Candes and T. Tao, “The dantzig selector: Statistical estimation when  $p$  is much larger than  $n$ ,” *The Annals of Statistics*, pp. 2313–2351, 2007.
- [17] J. A. Tropp and A. C. Gilbert, “Signal recovery from random measurements via orthogonal matching pursuit,” *IEEE Trans. on Information Theory*, vol. 53, no. 12, pp. 4655–4666, 2007.
- [18] D. Needell and J. A. Tropp, “Cosamp: Iterative signal recovery from incomplete and inaccurate samples,” *Applied and Computational Harmonic Analysis*, vol. 26, no. 3, pp. 301–321, 2009.

- [19] D. Needell and R. Vershynin, “Uniform uncertainty principle and signal recovery via regularized orthogonal matching pursuit,” *Foundations of computational mathematics*, vol. 9, no. 3, pp. 317–334, 2009.
- [20] D. L. Donoho, Y. Tsaig, I. Drori, and J.-L. Starck, “Sparse solution of underdetermined systems of linear equations by stagewise orthogonal matching pursuit,” *IEEE Trans. on Information Theory*, vol. 58, no. 2, pp. 1094–1121, 2012.
- [21] M. A. Davenport and M. B. Wakin, “Analysis of orthogonal matching pursuit using the restricted isometry property,” *IEEE Trans. on Information Theory*, vol. 56, no. 9, pp. 4395–4401, 2010.
- [22] M. A. Davenport, M. F. Duarte, Y. C. Eldar, and G. Kutyniok, “Introduction to compressed sensing,” *Preprint*, vol. 93, 2011.
- [23] S. D. Howard, A. R. Calderbank, and S. J. Searle, “A fast reconstruction algorithm for deterministic compressive sensing using second order reed-muller codes,” in *Information Sciences and Systems, 2008. CISS 2008. 42nd Annual Conference on*. IEEE, 2008, pp. 11–15.
- [24] L. Applebaum, S. D. Howard, S. Searle, and R. Calderbank, “Chirp sensing codes: Deterministic compressed sensing measurements for fast recovery,” *Applied and Computational Harmonic Analysis*, vol. 26, no. 2, pp. 283–290, 2009.
- [25] M. Akçakaya and V. Tarokh, “A frame construction and a universal distortion bound for sparse representations,” *Signal Processing, IEEE Transactions on*, vol. 56, no. 6, pp. 2443–2450, 2008.
- [26] F. Parvaresh and B. Hassibi, “Explicit measurements with almost optimal thresholds for compressed sensing,” in *Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on*. IEEE, 2008, pp. 3853–3856.
- [27] H. V. Pham, W. Dai, and O. Milenkovic, “Sublinear compressive sensing reconstruction via belief propagation decoding,” in *Information Theory, 2009. ISIT 2009. IEEE International Symposium on*. IEEE, 2009, pp. 674–678.
- [28] S. Sarvotham, D. Baron, and R. G. Baraniuk, “Sudocodes - fast measurement and reconstruction of sparse signals,” in *Information Theory, 2006 IEEE International Symposium on*. IEEE, 2006, pp. 2804–2808.
- [29] M. Bakshi, S. Jaggi, S. Cai, and M. Chen, “Sho-fa: Robust compressive sensing with order-optimal complexity, measurements, and bits,” in *Communication, Control, and Computing (Allerton), 2012 50th Annual Allerton Conference on*. IEEE, 2012, pp. 786–793.
- [30] F. Zhang and H. D. Pfister, “Compressed sensing and linear codes over real numbers,” in *Information Theory and Applications Workshop, 2008*. IEEE, 2008, pp. 558–561.
- [31] D. L. Donoho, A. Javanmard, and A. Montanari, “Information-theoretically optimal compressed sensing via spatial coupling and approximate message passing,” in *Information Theory Proceedings (ISIT), 2012 IEEE International Symposium on*. IEEE, 2012, pp. 1231–1235.
- [32] W. Xu and B. Hassibi, “Efficient compressive sensing with deterministic guarantees using expander graphs,” in *Information Theory Workshop, 2007. ITW’07. IEEE*. IEEE, 2007, pp. 414–419.

- [33] S. Jafarpour, W. Xu, B. Hassibi, and R. Calderbank, “Efficient and robust compressed sensing using optimized expander graphs,” *IEEE Trans. on Information Theory*, vol. 55, no. 9, pp. 4299–4308, 2009.
- [34] R. Berinde, A. C. Gilbert, P. Indyk, H. Karloff, and M. J. Strauss, “Combining geometry and combinatorics: A unified approach to sparse signal recovery,” in *Communication, Control, and Computing, 2008 46th Annual Allerton Conference on*. IEEE, 2008, pp. 798–805.
- [35] P. Indyk and M. Ruzic, “Near-optimal sparse recovery in the  $l_1$  norm,” in *Foundations of Computer Science, 2008. FOCS’08. IEEE 49th Annual IEEE Symposium on*. IEEE, 2008, pp. 199–207.
- [36] R. Berinde, P. Indyk, and M. Ruzic, “Practical near-optimal sparse recovery in the  $l_1$  norm,” in *Communication, Control, and Computing, 2008 46th Annual Allerton Conference on*. IEEE, 2008, pp. 198–205.
- [37] M. Charikar, K. Chen, and M. Farach-Colton, “Finding frequent items in data streams,” *Theoretical Computer Science*, vol. 312, no. 1, pp. 3–15, 2004.
- [38] P. Indyk, H. Q. Ngo, and A. Rudra, “Efficiently decodable non-adaptive group testing,” in *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms*. Society for Industrial and Applied Mathematics, 2010, pp. 1126–1142.
- [39] G. Cormode and S. Muthukrishnan, “Combinatorial algorithms for compressed sensing,” in *Structural Information and Communication Complexity*. Springer, 2006, pp. 280–294.
- [40] J. Haupt and R. Baraniuk, “Robust support recovery using sparse compressive sensing matrices,” in *Information Sciences and Systems (CISS), 2011 45th Annual Conference on*. IEEE, 2011, pp. 1–6.
- [41] A. C. Gilbert, M. J. Strauss, J. A. Tropp, and R. Vershynin, “One sketch for all: fast algorithms for compressed sensing,” in *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*. ACM, 2007, pp. 237–246.
- [42] A. C. Gilbert, Y. Li, E. Porat, and M. J. Strauss, “Approximate sparse recovery: optimizing time and measurements,” *SIAM Journal on Computing*, vol. 41, no. 2, pp. 436–453, 2012.
- [43] A. C. Gilbert, M. J. Strauss, J. A. Tropp, and R. Vershynin, “Algorithmic linear dimension reduction in the  $l_1$  norm for sparse vectors,” *arXiv preprint cs/0608079*, 2006.
- [44] E. J. Candès, J. Romberg, and T. Tao, “Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information,” *IEEE Trans. on Information Theory*, vol. 52, no. 2, pp. 489–509, 2006.
- [45] Y. Wu and S. Verdú, “Optimal phase transitions in compressed sensing,” *IEEE Trans. on Information Theory*, vol. 58, no. 10, pp. 6241–6263, 2012.
- [46] G. Reeves and M. Gastpar, “Sampling bounds for sparse support recovery in the presence of noise,” in *Information Theory, 2008. ISIT 2008. IEEE International Symposium on*. IEEE, 2008, pp. 2187–2191.

- [47] M. Gastpar and Y. Bresler, “On the necessary density for spectrum-blind nonuniform sampling subject to quantization,” in *Acoustics, Speech, and Signal Processing, 2000. ICASSP’00. Proceedings. 2000 IEEE International Conference on*, vol. 1. IEEE, 2000, pp. 348–351.
- [48] M. J. Wainwright, “Information-theoretic limits on sparsity recovery in the high-dimensional and noisy setting,” *IEEE Trans. on Information Theory*, vol. 55, no. 12, pp. 5728–5741, 2009.
- [49] S. Aeron, V. Saligrama, and M. Zhao, “Information theoretic bounds for compressed sensing,” *IEEE Trans. on Information Theory*, vol. 56, no. 10, pp. 5111–5130, 2010.
- [50] M. Akçakaya and V. Tarokh, “Shannon-theoretic limits on noisy compressive sampling,” *IEEE Trans. on Information Theory*, vol. 56, no. 1, pp. 492–504, 2010.
- [51] M. J. Wainwright, “Sharp thresholds for high-dimensional and noisy sparsity recovery using-constrained quadratic programming (lasso),” *IEEE Trans. on Information Theory*, vol. 55, no. 5, pp. 2183–2202, 2009.
- [52] T. T. Cai and L. Wang, “Orthogonal matching pursuit for sparse signal recovery with noise,” *IEEE Trans. on Information Theory*, vol. 57, no. 7, pp. 4680–4688, 2011.
- [53] A. K. Fletcher, S. Rangan, and V. K. Goyal, “Necessary and sufficient conditions for sparsity pattern recovery,” *IEEE Trans. on Information Theory*, vol. 55, no. 12, pp. 5758–5772, 2009.
- [54] W. Wang, M. J. Wainwright, and K. Ramchandran, “Information-theoretic limits on sparse signal recovery: Dense versus sparse measurement matrices,” *IEEE Trans. on Information Theory*, vol. 56, no. 6, pp. 2967–2979, 2010.
- [55] Y. Jin, Y.-H. Kim, and B. D. Rao, “Limits on support recovery of sparse signals via multiple-access communication techniques,” *IEEE Trans. on Information Theory*, vol. 57, no. 12, pp. 7877–7892, 2011.
- [56] A. Hormati, A. Karbasi, S. Mohajer, and M. Vetterli, “An estimation theoretic approach for sparsity pattern recovery in the noisy setting,” *arXiv preprint arXiv:0911.4880*, 2009.
- [57] T. J. Richardson and R. L. Urbanke, “The capacity of low-density parity-check codes under message-passing decoding,” *IEEE Trans. on Information Theory*, vol. 47, no. 2, pp. 599–618, 2001.
- [58] M. Finiasz and K. Ramchandran, “Private stream search at the same communication cost as a regular search: Role of ldpc codes,” in *Information Theory Proceedings (ISIT), 2012 IEEE International Symposium on*. IEEE, 2012, pp. 2556–2560.
- [59] M. G. Luby, M. Mitzenmacher, M. A. Shokrollahi, and D. A. Spielman, “Efficient erasure correcting codes,” *IEEE Trans. on Information Theory*, vol. 47, no. 2, pp. 569–584, 2001.
- [60] R. Pedarsani, K. Lee, and K. Ramchandran, “Phasecode: Fast and efficient compressive phase retrieval based on sparse-graph-codes,” *arXiv preprint arXiv:1408.0034*, 2014.
- [61] S. Kay, “A fast and accurate single frequency estimator,” *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 37, no. 12, pp. 1987–1990, 1989.
- [62] X. Li, S. Pawar and K. Ramchandran, “Sub-linear Time Support Recovery for Compressed Sensing using Sparse-Graph Codes,” *available online*: [http://www.eecs.berkeley.edu/~xiaoli/TR\\_CS\\_sublinear](http://www.eecs.berkeley.edu/~xiaoli/TR_CS_sublinear), 2014.